UNIVERSIDADE DE BRASÍLIA
DEPARTAMENTO DE ADMINISTRAÇÃO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO E CONTABILIDADE (FACE).

ANDRÉ DUARTE VERAS

Uma Proposta de Utilização de Redes Neurais Recorrentes na Previsão de Preços de Imóveis no Distrito Federal

Brasília, DF
2019

ANDRÉ DUARTE VERAS

Uma Proposta de Utilização de Redes Neurais Recorrentes na Previsão de Preços de Imóveis no Distrito Federal

Monografia apresentada ao Departamento de Administração como requisito parcial do título de Bacharel em Administração

Brasília, DF
2019

ANDRÉ DUARTE VERAS

Uma Proposta de Utilização de Redes Neurais Recorrentes na Previsão de Preços de Imóveis no Distrito Federal

Monografia apresentada ao Departamento de Administração como requisito parcial do título de Bacharel em Administração

Supervisor:
Prof. Peng Yaohao, M.Sc.

Brasília, DF
2019

FICHA

CATALOGRÁFICA

Veras, André Duarte

Uma Proposta de Utilização de Redes Neurais Recorrentes na Previsão de Preços de Imóveis no Distrito Federal / . – Brasília, DF, 2019. 82 p.

Monografia - Bacharelado — Universidade de Brasília - Departamento de Administração.

1. Redes Neurais Recorrentes. 2. Mercado Imobiliário. 3. Teoria da Decisão. I. Veras, André Duarte II. Universidade de Brasília.

DEDICATÓRIA

Dedico esse trabalho a minha família, que sempre esteve presente durante minha formação, especialmente minha mãe e meu pai.

RESUMO

O mercado imobiliário possui grande influência na economia de um país. Esse mercado movimenta diferentes áreas da sociedade brasileira e impulsiona o crescimento da economia sendo responsável por uma grande fatia do Produto Interno Bruto. Além disso, o crescimento das cidades também é influenciado pelos preços de seus imóveis. O investimento em imóveis é um dos principais investimentos dos brasileiros. A segurança que proporciona movimenta esse segmento e o torna um dos investimentos mais procurados no Brasil. Nesse contexto, surge a necessidade de compreender em quais regiões os imóveis vão valorizar nos próximos anos, de forma a maximizar o investimento realizado. Um dos principais objetivos deste trabalho é mostrar de que forma o uso de redes neurais recorrentes pode ser utilizada para prever os preços de imóveis na região do Distrito Federal nos próximos anos. Apesar de a região utilizada ser o Distrito Federal, essa abordagem pode ser utilizada em qualquer centro urbano brasileiro.

Nos procedimentos metodológicos, foi selecionado o histórico do preço médio mensal do imóvel como variável que influencia no valor futuro do imóvel. Essa informação foi utilizada na rede neural recorrente LSTM para previsão dos preços dos imóveis no ano seguinte. No trabalho em tela, os valores de crescimento médio dos preços dos imóveis foram agrupados no Distrito Federal por meio dos CEPs de cada região e a partir daí foi utilizado um mapa de calor para mostrar os resultados das regiões que tiveram maior crescimento. Também utilizou-se diferentes abordagens para analisar qual delas apresenta o melhor resultado. A primeira abordagem foi o modelo de *baseline Random Walk*, que serviu para comparar os modelos utilizados da rede LSTM. Posteriormente, foram utilizadas as redes LSTM, com uma abordagem de um passo a frente e de janela móvel. Dos modelos utilizados, o que obteve melhor resultado foi o de janela móvel. Analisando o mapa de calor do modelo que obteve os melhores resultados na validação de dados de teste, as regiões que apresentaram maior crescimento em dezembro de 2012 (1 ano após os últimos registros da amostra) foram, na ordem: São Sebastião e Lago Sul (84,29%), Sobradinho (64,41%) e Planaltina (44,22%).

ABSTRACT

The real estate market has great influence on a country's economy. This market impacts different areas of Brazilian society and drives the growth of the economy being responsible for a large share of the Gross Domestic Product. In addition, the growth of cities is also influenced by the prices of their real estate. Investment in real estate is one of the main investments in Brazil. The security this investment provides drives this segment and makes it one of the most sought after investments in Brazil. In this context, there is a need to understand in which regions real estat prices will increse in the coming years in order to maximize the investment made. One of the main objectives of this work is to show how the use of recurrent neural networks can be used to predict real estate prices in the Federal District in the coming years. Although the region used is the Federal District, this approach can be used in any Brazilian urban center.

In the methodological procedures, we selected the historical average monthly price of the property as a variable that influences the future value of the property. This information was used in the recurrent neural network LSTM to predict real estate prices for the following year. In tthis work, the values of real estate prices were grouped by region in the Federal District through the CEPs of each region and from there a heat map was used to show the results of the regions that had the highest growth. Different approaches have also been used to analyze which one has the best outcome. The first approach was the Random Walk baseline model, which served to compare the used models of the LSTM network. Subsequently, the LSTM networks were used with a step-by-step approach and moving window. Of the models used the one that obtained best result was the one of moving window. Analyzing the heat map of the model that obtained the best results in the validation of test data, the regions that showed the greatest growth in December 2012 (1 year after the last sample records) were in the order: São Sebastião and Lago Sul (84, 29 %), Sobradinho (64.41 %) and Planaltina (44.22 %).

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ADAM *Adaptive Moment Estimation*

ANFIS *Adaptive-Neuro-Fuzzy Inference Systems*

ARIMA *Autoregressive Integrated Moving Average*

ARM Análise de regressão Múltipla

BAR *Bayesian Autoregressive*

BPNN *Back Propagation Neural Network*

BVAR *Bayesian Vector Autoregressive*

CEF Caixa Econômica Federal

CEP Código de Endereçamento Postal

CFBP *Cascade Forward Back Propagation Neural Network*

CUB Custo unitário básico da construção civil

CSI *Cubic Spline Interpolation*

CPI *Condominium Price Index*

EEMD *Ensemble Empirical Mode Decomposition*

EQM Erro Quadrático Médio

FFBP *Feed Forward Back Propagation Artificial Neural Network*

FGV Fundação Getúlio Vargas

GP *Gaussian Process*

HDB *Singapore Housing and Development Board*

IBGE Instituto Brasileiro de Geografia e Estatística

IGPDI Índice Geral de PreçosDisponibilidade Interna

INA Nível de Atividade da Indústria

ITBI Imposto de Transmissão de Bens Imóveis Inter-Vivos

JOONE *Java Object Oriented Neural Engine*

LR *Linear Regression*

LSSVM *Least Squares Support Vector Machine*

LSTM *Long Short Term Memory*

MLP *Multilayer Perceptron*

MLS *Multiple Listing Services*

MRA *Multiple Regression Analysis*

MSE *Mean Squared Error*

OFHEO *Office of Federal Housing Enterprise Oversight*

OLS *Ordinary Least Squares*

PCA *Principal Component Analysis*

PCR *Principal Component Regression*

PLS *Partial Least Squares*

RFE *Recursive Feature Elimination*

RNA Rede Neural Artificial

RNF Redes Neuro-Fuzzy

RNN *Recurrent Neural Network*

RNR Rede Neural Recorrente

RMSE *Root Mean Square Error*

RW *Random Walk*

SPYDER *Scientific PYthon Development EnviRonment*

SVD *Singular Value Decomposition*

SVM *Support Vector Machine*

SVR *Support Vector Regression*

TODIM Tomada de Decisão Interativa e Multicritério

SUMÁRIO

1 INTRODUÇÃO

O mercado imobiliário pode ser considerado um setor-chave na economia brasileira. A aquisição de um imóvel é considerada uma importante decisão nos lares brasileiros. O sonho da casa própria é o principal objetivo de milhões de brasileiros. No entanto, a realização desse sonho requer um grande investimento por parte das famílias brasileiras. Além da escolha do tipo de imóvel e faixa de preço, muitas vezes o que se procura em um imóvel é sua localização e a rentabilidade futura. Logo, é importante, ao se investir em um imóvel, entender os aspectos de evolução do preço futuro daquele imóvel.

Este setor é composto dos imóveis e também das partes que realizam sua comercialização e locações. Além disso, as instituições financeiras são responsáveis pelo financiamento de grande parte desses empreendimentos e a construção civil por sua realização. Com isso, esse setor afeta diversos outros setores da economia gerando emprego e renda.

Entretanto, o mercado imobiliário possui um comportamento distinto dos mercados de outros bens. Os imóveis possuem características particulares e a avaliação e previsão de preços é uma tarefa complexa (GONZÁLEZ; FORMOSO, 2000).

A previsão e avaliação do preço dos imóveis sempre foi motivo de estudos, pois esse setor é um reflexo da economia do país. Logo, a previsão dos preços dos imóveis é importante tanto para as famílias brasileiras na hora de tomar a decisão de investir na aquisição de um imóvel quanto para economistas analisarem a situação econômica de uma região, cidade ou país.

Segundo González (1997), o mercado imobiliário pode ser influenciado por fatores públicos e privados. A condução econômica do governo tem influência sobre esse mercado, por isso é importante prever como se comporta o preço dos imóveis para a tomada de decisão de medidas macroeconômicas pelo governo e pela intervenção e regulamentação desse setor da economia.

Certamente, outros fatores também afetam o preço dos imóveis. Entre esses fatores, pode-se citar a localização e possibilidade de valorização futura. O valor da localização diz respeito à acessibilidade e características da vizinhança. Mesmo com diversos fatores que podem afetar o preço futuro do imóvel, o trabalho realizado utilizou apenas o histórico de preços dos imóveis

como variável explicativa para seu preço futuro.

Este trabalho tem como finalidade construir uma proposta de utilização de redes neurais recorrentes capazes de realizar previsões sobre preços de imóveis residenciais no Distrito Federal, por meio da construção de séries históricas. Esses dados sobre os preços de imóveis residenciais do Distrito Federal foram fornecidos pela Gerência Nacional e Acompanhamento de Risco e Crédito (Superintendência Nacional de Administração de Risco Corporativo) da Caixa Econômica Federal - CEF.

1.1 Problema da Pesquisa

Ao analisar o mercado imobiliário, a escolha da região do imóvel e sua futura valorização é um aspecto importante na decisão desse tipo de investimento. A escolha de um imóvel que proporciona a maximização de retorno do investimento é uma forma de permitir que novos negócios aconteçam e, consequentemente, que a economia nesse setor se mantenha aquecida.

O processo de estimar os preços dos imóveis é subjetivo, devido a dificuldade de se conhecer a fundo o comportamento do mercado em diferentes faixas de preço, locais e tipos de imóvel. Muitas vezes, é utilizada uma análise qualitativa do mercado, no qual as atributos valorativos dependem do ponto de vista do mercado e das perspectivas do vários intervenientes do processo.

Com o número elevado de vendas que ocorrem no mercado imobiliário, se torna cada vez mais demorado estimar os preços dos imóveis por meio da localização e características que possuem. A falta de informação dos agentes envolvidos dificulta ainda mais essa análise. Empresas do ramo imobiliário e pessoas que buscam adquirir imóveis perdem um tempo precioso estimando os valores desses imóveis. Muitas vezes, atributos como a qualidade da vizinhança e localização são de difícil determinação. Os imóveis são bens heterogêneos e existem diversas características a serem consideradas de forma simultânea para estimar o preço. Assim, caso essas características dos imóveis sejam medidas com imprecisões, isso pode acarretar graves prejuízos econômicos aos agentes envolvidos.

Além disso, muitas vezes as empresas realizam vendas de propriedades em localizações desconhecidas, o que torna ainda mais difícil realizar a pesquisa para estimar o preço de cada um desses imóveis. O preço de um imóvel é particular a uma região. O preço de um imóvel que está situado em uma região com graves problemas econômicos pode apresentar um comportamento diferente do preço de outras regiões. Isso dificulta o trabalho de imobiliárias no âmbito nacional.

Logo, a estimação de preços de imóveis de forma automatizada seria de grande benefício para economia, pois facilitaria o trabalho de empresas do ramo imobiliário e demais agentes que atuam nesse mercado. Além da automatização da previsão dos preços de imóveis, também é apresentado uma ferramenta visual que mostra o crescimento dos preços de imóveis em cada região do Distrito Federal. Essa ferramenta é composta por um gráfico de mapa de calor para identificar as regiões que tiveram maior crescimento comparativamente com outras regiões próximas e permite organizar, analisar e interpretar esses dados. Quando se tem um conjunto grande de informação para analisar, uma forma prática é utilizando um mapa de calor ou um mapa de escala de cor. Esse tipo de gráfico permite descobrir padrões e tendências, pois a informação é destacada com base em seus valores, comparativamente a outros do conjunto analisado. Essa ferramenta visual permite que gestores, investidores, analistas e administradores utilizem essa informação de maneira clara, rápida e intuitiva para tomada de diversas decisões cada um em sua respectiva área, além de facilitar a compreensão do comportamento do mercado de imóveis para todos envolvidos nesse mercado.

1.2 Objetivo Geral

Esta pesquisa tem como finalidade realizar previsões sobre os preços de imóveis no Distrito Federal, por meio do uso de uma série temporal com preços médios mensais e localização de imóveis transacionados entre os anos de janeiro de 1997 a dezembro de 2011 (180 meses), utilizando redes neurais recorrentes.

1.3 Objetivos Específicos

1. Levantamento da literatura recente no uso de aprendizado de máquina para previsão ou avaliação de preços de imóveis em trabalhos realizados no Brasil ou em trabalhos internacionais;
2. Discutir a evolução de modelos preditivos baseados em aprendizado de máquinas supervisionado;
3. Analisar os dados referentes aos preços de imóveis no Distrito Federal;
4. Tratar o conjunto de dados;
5. Deflacionar os preços dos imóveis por um índice de inflação adequado;

6. Utilizar redes neurais recorrentes para prever os preços dos imóveis do próximo ano a partir dos últimos dados de entrada;

7. Gerar um mapa de calor com dados georreferenciados do crescimento dos preços dos imóveis na região do Distrito Federal;

1.4 Motivação/Justificativa

O mercado imobiliário é parte fundamental na economia brasileira. O estudo da previsão de preços dos imóveis é de grande importância. Ele visa auxiliar as pessoas na tomada de decisão em relação ao preço justo e localização de imóveis, que resultará em um preço desejado no futuro. Muitas vezes essas transações financeiras de imóveis são marcadas por assimetria de informação entre compradores, intermediários e vendedores. Por isso, existe a necessidade de coleta e processamento de informações de forma a atenuar essa assimetria. Muitas vezes a coleta dessas informações pode ser dispendiosa e ineficiente.

Com o advento de um mundo globalizado, as mudanças são rápidas e as empresas precisam responder a essas mudanças em um velocidade maior. Com isso, essas empresas buscam técnicas e ferramentas que auxiliem na tomada de decisões nos negócios.

Um método eficiente para previsão de preços pode permitir que as transações dos imóveis sejam justas para ambas as partes, e reduzir o risco de manipulação pelos agentes imobiliários.

Além disso, a previsão e avaliação dos preços de imóveis poderiam contribuir para estimar o valor de impostos que incidem sobre a propriedade urbana, como o IPTU (Imposto Predial e Territorial Urbano e ITBI - Imposto sobre Transações de Bens Imobiliários). A tributação de propriedades urbanas em muitas cidades do Brasil é calculada pela medida de área e localização da propriedade (BRONDINO, 1999). Esse cálculo desconsidera outros fatores e pode ser ineficiente em alguns casos.

O planejamento de financiamento imobiliário de instituições financeiras poderia utilizar as previsões de preços dos imóveis para embasar projeções de transações de financiamento. Essas transações de financiamento dependem do valor dos imóveis (GONZÁLEZ, 2002).

Outra aplicação para a previsão de preços por meio das redes neurais, seria medir a influência de investimentos e projetos públicos em uma determinada região da cidade com base em um histórico de ouros investimentos similares realizados, no preço dos imóveis dessa região. O que poderia ser de grande valia para órgãos públicos na análise do planejamento urbano. Além disso, é possível prever os custos de desapropriação de imóveis para programas habitacionais

do governo ou reforma agrária. Logo, um modelo de previsão de preços poderia ser utilizado na definição de planos diretores e estudos de viabilidade econômica de novas construções.

Este trabalho tem como principal justificativa construir uma proposta de utilização de aprendizagem de máquinas (em inglês: "*machine learning*"). O aprendizado de máquina é o uso de dados para o aprendizado. No trabalho proposto, mais especificamente, foi utilizado Redes Neurais Recorrentes na previsão de preços de imóveis no Distrito Federal. Entre as vantagens dessa proposta seria reduzir o tempo de previsão de preços, maximizar o retorno em investimentos de imóveis, evitando perdas e má alocação dos recursos disponíveis.

O uso de aprendizagem de máquina no setor imobiliário se torna cada vez mais importante no campo preditivo. Na medida em que informações imobiliárias, como preços de comercialização estão cada vez mais disponíveis e em grandes quantidades, possibilita o desenvolvimento de modelos preditivos e reconhecimento de padrões. Logo, a aprendizagem de máquina permite lidar com grandes quantidades de informação, automatizar a carga dessas informações e, posteriormente, visualizar o relacionamento das informações obtidas com a finalidade de auxiliar na tomada de decisões, como por exemplo, de comprar ou vender um imóvel.

Na administração, aprendizagem de máquina está cada vez mais presente na tomada de decisões das empresas. Essas empresas tentam absorver conhecimento e utilizar técnicas e ferramentas modernas de gestão. Muitas delas investem em tecnologia e inovação, principalmente no uso de aprendizagem de máquina com o objetivo de aumentar a produtividade e eficiência, tanto operacional, quanto gerencial. No campo gerencial, com as inovações tecnológicas é possível obter informações de maneira mais rápida e fácil. Com isso, o processo de tomada de decisões nos diversos níveis da organização pode ser aperfeiçoado e contribuir para a gestão das empresas. Além disso, esse campo de estudos pode reduzir custos nas empresas, aumentar a venda de produtos e serviços e possibilitar a criação de novas estratégias para as empresas.

2 REFERENCIAL TEÓRICO

2.1 Considerações Iniciais

Este capítulo trará uma revisão da literatura por meio dos principais artigos sobre mercado imobiliário, avaliação e previsão de preços de imóveis nacional e internacional. A avaliação e previsão dos preços serão analisadas em relação ao método utilizado e as variáveis consideradas. A pesquisa realizada foi concentrada em artigos e dissertações que utilizaram técnicas de aprendizado de máquina obtidos utilizando o Google Acadêmico, a partir de 1997 até o ano de 2018.

2.2 Nacional

Nesta seção são listados todos os trabalhos realizados sobre o tema de previsão ou avaliação de mercado imobiliário no Brasil.

2.2.1 Análise de Regressão Linear Múltipla

O processo de avaliação e previsão de preços de imóveis por meio da análise de regressão linear múltipla envolve a estimação de parâmetros que representam o comportamento do mercado imobiliário. Esses parâmetros são as variáveis independentes que se relacionam com a variável dependente, o valor do imóvel.

Os primeiros trabalhos com métodos de regressão linear foram desenvolvidos por Dantas e Cordeiro (1988). Neste trabalho, foi utilizado um modelo clássico de regressão para estudar os preços dos imóveis utilizando como variável dependente, o valor unitário à vista do imóvel em metro quadrado, e como variáveis independentes, as características desse imóvel como a localização, o número de quartos e outros atributos. Esse modelo, porém, não fornece uma visão total do mercado e de outros fatores que influenciam o preço de um imóvel.

Em seguida, González (1997) analisou a evolução dos preços de aluguéis na cidade de Porto Alegre, de antes do Plano Real até 1996. No estudo realizado o foco era como a estabilização econômica influenciou o preço dos aluguéis e a oferta de apartamentos residenciais. Para isso, foram coletados dados de 980 apartamentos residenciais diretamente do mercado e dados de pesquisas mensais do Sindicato de imobiliárias de Porto Alegre. Os dados mensais do Sindicato não são conferidos e são coletados dos classificados de jornais. Também não se sabe se a transação ocorreu de fato. Os dados foram analisados por meio de modelos hedônicos de preços. Com o uso do modelo hedônico e o conjunto de dados também foi possível fazer algumas previsões. No preço dos aluguéis foi descontada a inflação, medida pelo IGP-DI.

Segundo Balarine (1997), as variáveis macroeconômicas como renda, nível das taxas de juros, disponibilidade de financiamentos e outras variáveis também afetam a formação dos preços dos imóveis. Para esse estudo foram utilizados dados de variáveis socioeconômicas confiáveis da cidade de Porto Alegre. A variável dependente utilizada foi o preço médio das habitações de apartamentos de dois quartos na cidade de Porto Alegre, obtidas em uma revista de imóveis. As variáveis independentes foram o estoque habitacional (número de domicílios em Porto Alegre com ligação de energia elétrica, renda real (renda real per capita no município, estimada com base no produto interno bruto do estado Rio Grande do Sul), grau de concentração de renda (índice de Gini apurado pelo IBGE), população (total da população de Porto Alegre), consumo (consumo de energia elétrica residencial), aluguéis (preço médio mensal dos aluguéis de apartamentos de dois quartos), inflação (IGP-DI divulgado pela FGV), custos da construção (custo unitário básico levantado para Porto Alegre pelo Sindicato das Indústrias da Construção Civil do Estado do Rio Grande do Sul), juros (taxa média dos juros reais das cadernetas de poupança), financiamentos (número de unidades habitacionais financiadas). Apesar das evidências empíricas confirmarem o objetivo desse trabalho de que a formação de preços podem ser explicados por análises macroeconômicas, os modelos utilizados são simplificações da realidade e não contemplam todos os fatores que afetam a formação dos preços. Porém a análise macro desses imóveis pode agregar conhecimentos relevantes à avaliação de imóveis, juntamente com a análise micro. Logo, é possível ter uma visão mais abrangente da formação do preço dos imóveis.

Machado e Heineck (1998) apresentaram um estudo econométrico utilizando análise de séries temporais para descrever a evolução do preço em apartamentos residenciais de três quartos na cidade de Florianópolis, em relação a variáveis dependentes econômicas e de mercado. O estudo teve como foco gerar uma equação de regressão múltipla que fosse capaz de prever preços de apartamentos residenciais. Em seguida, foi criado por meio de software estatístico, um modelo para descrever a influência das diversas variáveis sobre os preços dos

imóveis. O modelo utilizado foi o dos Mínimos Quadrados Ordinários (OLS). A variável dependente utilizada foi a variação dos preços dos apartamentos. As variáveis independentes do modelo foram o valor do aluguel dos imóveis, o custo unitário básico da construção (CUB) e nível de atividade industrial do país (INA). Os dados foram coletados entre outubro de 1994 e novembro de 1997, em informes semanais de jornais da cidade de Florianópolis. Para os dados coletados, foi utilizada uma média aritmética. Com a estabilidade da moeda, foi considerado que não era necessário um processo de deflacionamento no preço dos apartamentos.

González e Formoso (2000) mostraram as limitações do uso de análise de regressão múltipla (ARM) na avaliação de preços de imóveis. As dificuldades estão relacionadas pelas peculiaridades do mercado e de dois principais problemas: a autocorrelação espacial e a determinação da forma funcional. A autocorrelação ocorre quando os estimadores obtidos por Mínimos Quadrados são não-viesados, ineficientes e os modelos não são plenamente válidos. Logo, a autocorrelação espacial é um dos principais problemas nas análises de preços de imóveis. A determinação da forma funcional diz respeito à formatação do modelo, ou seja, os pressupostos de linearidade da equação e as variáveis utilizadas no modelo. Também é um problema de difícil solução a determinação de quais variáveis incluir no modelo e em que formato essa variável deve estar. O trabalho sugere verificar as variáveis por meio de tentativa e erro. Por meio da experimentação, as funções podem ser ajustadas e as variáveis transformadas. Com exemplo dessas transformações pode-se citar logaritmos, inversas ou potências. Nesses casos deve-se ter cuidado ao escolher essas transformações para não inviabilizar o modelo.

Silva, Nali e Marote (2009) apresentaram um estudo para avaliação de imóveis rurais irrigáveis por meio de regressão linear múltipla. A variável dependente é o valor do imóvel e as independentes são a data da negociação, nível de infraestrutura, elasticidade da oferta, área do imóvel, percentual de área irrigável e área equivalente de produção vegetal. A regressão linear múltipla permite uma maior precisão nas avaliações em casos de desapropriação por utilidade pública, na construção de obras públicas ou por interesse social. Também contribui nas avaliações de casos como a reavaliação de ativos de empresas, partilha oriunda de herança, meações, divórcios, lançamentos de impostos como o ITBI e ITR e outras contribuições. O estudo utilizou 32 amostras de pesquisas de preços coletadas pelo INCRA na microrregião de Petrolina em Pernambuco. A pesquisa das transações de vendas de imóveis foi realizada em cartórios e corretoras com a confirmação com o comprador e vendedor do imóvel. Os preços dos imóveis, por serem sempre positivos são transformados para a escala logarítmica, uma vez que o logaritmo do preço abrange a reta real. Os resultados do estudo apresentaram um bom ajustamento aos dados observados, sendo aceitas as hipóteses estatísticas que permitem sua validação.

Araújo et al. (2012) realizaram uma pesquisa de regressão linear múltipla para avaliação de imóveis residenciais urbanos na cidade de Bonito no Mato Grosso do Sul. Foram utilizados 27 observações de imóveis residenciais obtidas em sítios de imobiliárias da cidade de Bonito. As variáveis independentes dos imóveis utilizadas foram: área total do imóvel, consumo de energia, distância da escola, acessibilidade, idade do imóvel, dormitórios, meio ambiente, região homogênea, zona fiscal, padrão de entrada, classificação, conservação, garagem, suíte, dependência de empregada, elevador, pólos de valorização, área útil, área do terreno, número de vagas na garagem e localização. A variável de localização assumiu o valor 0 para imóveis fora do centro e 1 para imóveis situados no centro. Nesse trabalho foi utilizada a técnica de *Ridge Regression* para evitar a multicolinearidade. O método dos mínimos quadrados foi utilizado para estimar a regressão. Os critérios utilizados para seleção das variáveis independentes foram Seleção Progressiva e Seleção Regressiva. O modelo adotado apresentou erros percentuais variando de 2,43 a 3,91%.

2.2.2 Redes Neurais Artificiais

Os trabalhos em avaliação e previsão de preços no mercado imobiliário evoluíram e passaram a utilizar técnicas de redes neurais artificiais.

Um dos primeiros trabalhos nesse tema foi de Brondino (1999), que identificou as principais variáveis que interferem no valor das propriedades e propuseram o uso de redes neurais artificiais para avaliar e estudar a influência de uma medida de acessibilidade (distância ao centro da cidade) no valor de terrenos urbanos. Foram utilizadas variáveis de natureza espacial (distância ao centro da cidade) e atributos físicos dos imóveis. Também foram comparados dois métodos de avaliação: as Redes Neurais Artificiais (RNA) e o modelo de regressão múltipla. Foram analisadas propriedades nas cidades de Araçariguama e São Carlos no estado de São Paulo. Na cidade de Araçariguama a amostra foi da totalidade de terrenos vazios situados em seis bairros. Já na cidade de São Carlos, a amostra foi dimensionada com base no valor por metro quadrado. Os dados foram obtidos da prefeitura das respectivas cidades. Os terrenos escolhidos tiveram seus mapas digitalizados para levantar as características espaciais. O conjunto de dados foi dividido em três grupos distintos: treinamento (50% dos dados), validação (25%) e teste (25%). Essa divisão foi feita por três vezes para cada cidade. Alguma das variáveis utilizadas para os lotes de Araçariguama foram: forma (valor binário de regular ou irregular), testada (testada principal do terreno em metros), área do terreno em metros quadrados, existência de muro e calçada e situação dentro da quadra. E a última variável que era o menor caminho percorrido até o centro por meio da malha viária, foi obtida pelo software TransCAD. Os preços dos imó-

veis também foram obtidos na prefeitura das respectivas cidades. Em Araçariguama a amostra é de 157 lotes. Algumas das variáveis utilizadas para os lotes de São Carlos foram: área do terreno em metros quadrados, testada principal do terreno em metros, topografia (declividade), forma (regular ou irregular), situação dentro da quadra, edificações na quadra, água (existência de rede de água tratada), esgoto (existência de rede de esgoto), luz (existência de rede de energia elétrica), asfalto (existência de pavimentação), muro (existência de muro) e existência de calçada. Além disso, também foi utilizada a medida de acessibilidade (distância até o centro da cidade por meio da malha viária). Os resultados obtidos mostraram que o modelo de redes neurais obteve um erro menor que o modelo de regressão. Além disso, o uso de uma variável de natureza espacial contribui para o modelo proposto.

Em seguida, Nghiep e Al (2001) apresentaram uma comparação na previsão de preço de imóveis entre Redes Neurais Artificiais (RNA) e Análise de Regressão Múltipla (ARM). As comparações são feitas usando dois modelos de dados. Nesses modelos são alterados o tamanho dos dados de exemplo, a especificação funcional e critérios de avaliação. As redes neurais tiveram um desempenho melhor que a análise de regressão múltipla quando um tamanho grande da base de dados foi utilizado. Para o estudo foi utilizado 3.906 observações. Desse total foram tiradas amostras variando de 506 até 1506 observações. A rede neural com *Back Propagation* e *Feed Forward* foi escolhida entre várias arquiteturas de redes neurais testadas por ter melhor desempenho. As redes neurais artificiais possuem a habilidade de aprender a reconhecer padrões de dados complicados sem a necessidade de serem programadas com regras preconcebidas. Logo, podem ser aplicadas com pouco conhecimento estatístico do conjunto de dados. Porém alguns problemas surgem como, por exemplo, definir o número de camadas ocultas, o número de neurônios e a escolha do conjunto de treinamento e de validação. O uso de regressão múltipla também cria problemas como, por exemplo, a definição da forma funcional, a falta de especificação, a não linearidade e multicolinearidade. Para a previsão de valor do imóvel foi constatado que a idade do imóvel e a metragem não possuem uma relação linear com o preço do imóvel. No estudo foram realizadas 108 comparações. Ao se utilizar uma especificação do modelo funcional mais complexa o conjunto de dados treinado deve ser aumentado para que as redes neurais tenham um desempenho melhor que o método de regressão múltipla. Nos casos em que se utilizou um conjunto pequeno de dados, a ARM teve um desempenho superior às RNA.

González (2002) propõe técnicas alternativas ao modelo hedônico de preços para desenvolver modelos preditivos para o preço de imóveis baseado em inteligência artificial. Essa nova abordagem para avaliação dos preços, consiste na formação de uma base de dados previamente tratada e preparada e por meio de conjunto de técnicas para selecionar casos para gera-

ção de modelos preditivos. Inicialmente, os dados foram organizados por meio de técnicas de regressão e de redes neurais para a escolha de informações relevantes. Além disso, o algoritmo de vizinhança próxima foi utilizado para estimar os valores para os dados que estavam faltando ou apresentavam erros. O modelo preditivo utilizado incluiu as técnicas de regressão com superfícies de resposta, modelos aditivos generalizados, redes neurais usando lógica difusa e sistemas de regras difusas obtidos com algoritmos genéticos. Esses modelos foram comparados com o modelo de regressão múltipla. No estudo, dados das declarações do Imposto sobre Transmissão de Bens Imóveis (ITBI) foram fornecidos pela Prefeitura Municipal de Porto Alegre, com mais de 30 mil transações de apartamentos residenciais. Os modelos apresentaram uma melhor precisão que a regressão múltipla. O modelo com melhor desempenho foi o de sistema de regras difusas. O processo proposto começa com o tratamento dos dados. Inicialmente foram identificados os casos irrelevantes, considerados como *outliers*. Esses dados foram removidos, pois prejudicam a análise e o resultado. Posteriormente, foram selecionados os casos em paralelo com os atributos de forma incremental e construindo modelos aprimorados. Além disso, foram sendo identificados novos *outliers*. No trabalho, foram propostas três modelos de avaliação. São eles os modelos gerais, modelos destinados à avaliação de massa e modelos de avaliação individual. Esses três formatos de avaliação tiveram seu desempenho comparados. A técnica de sistemas baseados em regras difusas evolucionárias teve desempenho superior à regressão. Porém, os resultados foram similares e todas as técnicas podem ser aproveitadas, tanto na avaliação individual quanto na coletiva. A utilização de mais de uma técnica permite a verificação empírica segura dos efeitos dos dados sobre a regressão. Para os modelos coletivos o coeficiente de dispersão foi de 18% e para o modelo de avaliação individual o coeficiente foi de 7,4%.

Neto (2004) apresentou um estudo com três metodologias para estimação do valor de mercado de imóveis por meio de Redes *Neuro-Fuzzy* (RNF) ou Sistemas Nebulosos e Redes Neurais Artificiais (RNA). Posteriormente, os resultados foram comparados com uma regressão linear múltipla por estimadores de mínimos quadrados. Primeiramente, foi realizado um processo de seleção e análise dos dados para treinamento. Depois, seguiram-se o pré-processamento, o processo de treinamento das redes e o processo de validação e análise dos resultados. O estudo considera que o mercado imobiliário apresenta comportamentos não lineares e que necessitam de soluções analíticas ou numéricas. Além disso, a avaliação de imóveis envolve a estimação de diversos parâmetros populacionais. Essas variáveis são as variáveis independentes ou variáveis de entrada do modelo, e se relacionam com o valor do imóvel. O valor do imóvel é a variável dependente ou a variável de saída. As variáveis de entrada e de saída se relacionam de forma não linear. No trabalho foram utilizadas nove variáveis independentes

(nível/elevador, setor, total de vagas, área coberta, número de dormitórios, número de sanitários, equipamentos, padrão de acabamento e estado de conservação). Foram selecionados 172 apartamentos no mercado imobiliário da cidade de Belo Horizonte, no estado de Minas Gerais. Esses apartamentos se encontram em regiões distintas do município de Belo Horizonte e suas características físicas intrínsecas e extrínsecas foram descritas de acordo com as nove variáveis independentes. Foram retirados 22 casos para validação dos modelos matemáticos. Os 150 restantes foram utilizados na regressão linear múltipla. As modelagens matemáticas por meio das Redes *Neuro-Fuzzy* e das Redes Neurais Artificiais tiveram seus dados previamente tratados e usados no treinamento dessas redes. A estrutura das Redes *Neuro-Fuzzy* está baseada em conjuntos nebulosos e são implementados em sistemas computacionais que dão origem aos sistemas de inferência nebulosos ou ANFIS (*Adaptive-Neuro-Fuzzy Inference Systems*) que combinam regras do tipo Se-Então com estruturas de RNA. O modelo ANFIS utilizou 112 nós, 50 parâmetros lineares, 90 parâmetros não lineares e 5 regras nebulosas. Foram realizados cinco seleções para treinamento. As redes neurais artificiais utilizadas eram multicamadas com o aprendizado do tipo supervisionado, baseado no sistema de retro propagação de erro. A primeira camada da rede neural artificial recebe as entradas externas e a camada de saída é responsável pela geração da resposta. As camadas do meio são chamadas de camadas ocultas ou escondidas (*Hidden Layers*). O número de neurônios da camada oculta era igual a 2N + 1 (19 neurônios). Para cada conjunto padrão os pesos da rede foram ajustados para minimizar a diferença entre as saídas da rede e o resultado desejado. As RNAs com multicamadas tiveram o melhor desempenho e se aproximaram mais dos valores de mercado.

Baptistella (2005) utilizou redes neurais artificiais na estimação dos preços de imóveis urbanos na cidade de Guarapuava no estado do Paraná. Foram utilizados dados de 300 imóveis residenciais (casas e apartamentos) obtidos do cadastro imobiliário fornecido pelo setor de planejamento da Prefeitura de Guarapuava. Inicialmente, foram utilizados treze atributos: bairro, setor, pavimentação, esgoto, iluminação pública, área do terreno, pedologia, topografia, situação, área edificada, tipo, estrutura e conservação. Após a coleta dos dados, foi utilizada a técnica da Análise das Componentes Principais (*PCA – Principal Component Analysis*) para reduzir e transformar as variáveis originais em nove fatores. A RNA utilizada foi do tipo *Feed Forward* com apenas uma camada oculta. A amostra de residências foi posteriormente reduzida para 256 unidades após uma limpeza nos dados. Os apartamentos com as mesmas características foram considerados como apenas um apartamento. A RNA de múltiplas camadas utilizou o algoritmo *Back-Propagation* padrão com o método de gradiente decrescente. A Análise das Componentes Principais foi utilizada para reduzir os números de variáveis com a finalidade de facilitar a interpretação. Com o uso dessa técnica, o número de variáveis passou de treze para

nove componentes principais. A RNA foi criada com a topologia MLP (*MultiLayer Perceptron*) **FeedForward** com uma camada de entrada com nove neurônios, conforme as variáveis obtidas na Análise das Componentes Principais, uma camada oculta com um número de neurônios variando de zero a doze, e uma camada de saída, com um único neurônio que fornecerá o valor do imóvel. A função de ativação utilizada para as camadas ocultas e de saída foi a função não linear sigmoidal (logsig). Essa função assume valores no intervalo de 0 a 1. O treinamento utilizou o algoritmo LM. Além disso, o desempenho foi avaliado utilizando o erro quadrático médio (*MSE – Mean Squared Error*) e a raiz quadrada do erro quadrático médio (*RMSE – Root Mean Squared Error*) no cálculo do erro da rede. Para o experimento, a amostra foi dividida em dois grupos. O primeiro grupo de treinamento contava com uma amostra de 170 imóveis e o segundo grupo de teste com 86 imóveis. Nos dois grupos foi mantida a mesma proporção de valores dos imóveis. Foram realizados 50 treinamentos com variação dos pesos iniciais e o critério de parada para cada treinamento. O software utilizado para os cálculos das redes neurais foi o *Matlab 6.5*. Na análise de regressão foi utilizado o software *Excel* com validação dos resultados por meio do software *Statistica v.5*. O modelo de redes neurais apresentou percentual de acerto das previsões de 89,60% contra 73,24% da regressão.

Moreira, Silva e Fernandes (2010) pesquisaram sobre metodologias de avaliações e preços de imóveis existentes e desenvolveram uma nova abordagem baseada na utilização de Redes Neurais Artificiais (RNA) e do método de Tomada de Decisão Interativa e Multicritério (TODIM). O resultado da técnica TODIM foi utilizado com uma das entradas da RNA e as outras entradas correspondem às características do imóvel. A modelagem da rede foi realizada no *framework JOONE* (*Java Object Oriented Neural Engine*) para executar aplicações em redes neurais. Além disso, a rede criada foi do tipo MLP (*MultiLayer Perceptron*), com conexões *Feedforward* e o algoritmo de treinamento utilizado foi o *Back Propagation* com supervisão. O trabalho desenvolveu uma avaliação na qual, inicialmente, é realizada uma carga de dados no sistema, depois é aplicado o método TODIM na base de dados nas alternativas, e, em seguida, as alternativas são ordenadas e um valor médio de aluguel para cada alternativa é calculado. Após a aplicação do TODIM, ocorre o treinamento da rede neural com as características das alternativas juntamente com o valor médio estimado pelo método TODIM. Assim, para avaliar um imóvel, são utilizadas as características do imóvel e os dados são aplicados na entrada da rede neural que apresente como resultado o valor do aluguel. A rede neural utilizada foi de múltiplas camadas, com 13 neurônios na camada de entrada, onde cada neurônio representa uma característica do imóvel: valor venal, valor de condomínio, valor do IPTU, peso da localização, peso do tipo, número de quartos, número de suítes, número de banheiros, mobiliado, número do andar, número de vagas de garagem e o valor médio do TODIM. A camada de saída

possuía 4 neurônios capazes de representar a quantidade de faixas de valores de aluguéis que o resultado disponibiliza. Além disso, a camada oculta possuía 180 neurônios. A base de dados de treinamento contou com 600 observações, enquanto a de teste e validação possuía 200 registros. Nesse trabalho, foi implementado um protótipo que permitiu avaliar diferentes tipos de imóveis. Em 30 avaliações realizadas, 21 avaliações (70%) ficaram dentro da faixa de aluguel sugerida pela RNA que continha o valor de aluguel real dentro da mesma.

2.3 Internacional

No âmbito internacional também foram desenvolvidos muitos trabalhos na área de previsão e avaliação de preços de imóveis.

2.3.1 Análise de Regressão Linear Múltipla

Na área de Regressão Linear, Selim (2011) examinou os principais atributos que afetam os preços de imóveis e de aluguéis na Turquia no modelo de preços hedônicos. Foram utilizados dados do levantamento *Household Budget Survey Data* (Instituto de Estatística Turco) de 2014. Os dados utilizados nesse trabalho incluem todas as regiões da Turquia e áreas urbanas e rurais. Os principais atributos que afetam os preços dos aluguéis são o tipo de casa, tipo de prédio, número de quartos, tamanho, e outras características estruturais como sistema de água, piscina e sistema de gás. O modelo hedônico emprega técnicas de regressão múltipla em grandes conjuntos de dados e avalia as características utilitárias dos bens. Porém, essa abordagem pode ser problemática se forem considerados *outliers*, não linearidade, dependência espacial ou de outras variáveis e também a possibilidade de descontinuidades. As redes neurais podem ser consideradas como uma abordagem mais flexível em relação à regressão. O tamanho da amostra é de 5.741 com 46 atributos. Além disso, foi utilizado a forma funcional semi-logarítmica. Mais especificamente, o logaritmo natural do preço do imóvel é tratado como uma variável dependente. Nos resultados do trabalho a heteroscedasticidade está presente. Esse é um problema comum nas equações de modelos de preços hedônicos.

2.3.2 Métodos de Kernel

Saunders, Gammerman e Vovk (1998) realizaram uma regressão não linear por meio da construção de uma função de regressão linear em um espaço de dimensão maior para determinar os preços dos imóveis na cidade de Boston, nos Estados Unidos. No estudo foi utilizado

funções *Kernel* para evitar a maldição da dimensionalidade. Nesse trabalho são discutidas funções *Kernel* construídas por meio de decomposição ANOVA. O trabalho também apresenta um algoritmo de regressão que é uma combinação entre a versão dual da regressão de *Ridge* e a decomposição ANOVA. O trabalho utilizou um conjunto de dados de preços de imóveis da cidade de Boston nos Estados Unidos. O conjunto de dados é formado por 506 casos e 13 variáveis independentes. Dessas variáveis, 12 são variáveis contínuas e uma variável é binária. Com essas observações é possível determinar o preço médio, em milhares de dólares, de imóveis em uma determinada região de Boston. As variáveis contínuas representam a localização e características econômicas e estruturais dos imóveis. Os preços variavam de $5.000 a $50.000. O conjunto de dados foi particionado em um conjunto de treinamento (401 casos), conjunto de validação (80 caso) e um conjunto de testes (25 casos). Essa divisão foi feito de forma aleatória por 100 vezes, de forma a realizar 100 testes nos dados. Para os testes foram utilizados um *Kernel* de *Splines*, um *Kernel* com decomposição ANOVA e um *Kernel* Polinomial.

Caplin et al. (2008) demonstraram o padrão sistemático de erros associados ao uso da metodologia de repetição de valores de vendas para compreender o valor dos imóveis. Em alguns anos os preços das casas foram subestimados e em outros foram superestimados. O estudo apresentou um modelo para previsão de preços de imóveis com técnicas de aprendizado de máquina. Esse modelo pode corrigir previsões que possuem origem geográfica e remove os padrões sistemáticos de erros. No estudo foi utilizado 1,5 milhões de dados de transação de imóveis na cidade de Los Angeles no estado da Califórnia nos Estados Unidos. Esses dados foram coletados ao longo de 25 anos. Da base inicial de 1,5 milhões de registros foram selecionados 3.630.759 transações com os seguintes atributos: preço de venda, a data de registro e a data de venda. Em seguida, os dados passaram por um tratamento. As transações de um mesmo imóvel, em uma mesma data, foram unificadas e transações de um mesmo imóvel, com menos de sete meses, foram removidas. Após a limpeza, sobraram 1.550.451 registros de transações. O estudo propôs um modelo de regressão local para encontrar o vizinho mais próximo de cada imóvel no conjunto de testes. Em seguida, são utilizadas funções *Kernel* espaço-temporal para suavizar o preço não observável. O modelo é testado em 18 diferentes períodos de testes que foram construídos baseados no mês da transação do imóvel. Os períodos foram coincidentes com períodos de crescimento e declínio no mercado imobiliário em Los Angeles. O modelo também utiliza coordenadas GPS com uma variável de entrada. O modelo proposto apresenta desempenho melhor do que o índice *Case-Shiller*, que utiliza a metodologia de repetição de preços.

Yoo, Im e Wagner (2012) utilizaram abordagens de aprendizado de máquina para a seleção de variáveis independentes do modelo hedônico de preços e o modelo de preços de imóveis.

Dois métodos de regressão baseados em regras de aprendizagem de máquina (*Cubist* e *Random Forest*) foram comparados com o modelo tradicional de regressão por Mínimos Quadrados Ordinários (*OLS – Ordinary Least Squares*). Foram utilizados 4.469 dados da cidade de Onondaga nos Estados Unidos. Para análise desses dados foram utilizadas condições de vizinhança com 100 metros e 1 quilômetro de raio. Os resultados mostraram que o método *Random Forest* obteve o melhor desempenho em relação à acurácia.

Pow, Janulewicz e Liu (2014) analisaram os preços dos imóveis em Montreal no Canadá por meio de técnicas de aprendizagem de máquina. Os dados de 25 mil imóveis foram obtidos de dois sítios de imobiliárias. No estudo, tanto o preço ofertado quanto o vendido foram estimados. Algumas das variáveis independentes dos imóveis utilizadas no estudo foram a localização geográfica, a área útil e número de quartos. Além disso, foram utilizadas outras características dos imóveis, como a proximidade do imóvel de um posto policial e do corpo de bombeiros. Foram analisados e comparados os modelos de regressão linear, regressão de suporte vetorial, *k-Nearest Neighbours* (k-NN) e árvore de regressão/regressão de *Random Forest*. O preço de oferta obteve um erro de 9,85% usando k-NN e algoritmos de *Random Forest*. O preço final de venda pode ser previsto com um erro de 2,3% utilizando regressão de *Random Forest*. A estimativa desses dois preços do imóvel é importante, pois fornece um sistema automático de previsão independente de interesses das partes envolvidas na transação. Esse sistema pode também buscar no mercado imobiliário, imóveis que possuem preços superestimados ou subestimados. E com isso, esse sistema pode constituir poderosa ferramenta em investimentos no mercado imobiliário. A base de dados possui 25 mil observações e 130 variáveis independentes. Entre as variáveis independentes estão características dos imóveis, localização geográfica e características sóciodemográficas da região. Entre as variáveis sóciodemográficas, podem ser citadas a densidade populacional, renda média e o tamanho da família. Além disso, foram incluídas variáveis de proximidade a postos policiais e corpo de bombeiros. Os *outliers* foram determinados por inspeção humana após análise da distribuição de valores. Em seguida, foram corrigidos ou removidos. Os dados que apresentavam alguma variável faltando também foram removidos da base de dados. Imóveis com preços abaixo de 10.000 também não foram incluídos na análise, pois provavelmente eram erros. Além disso, imóveis superiores a 4 vezes a faixa interquartil também foram considerados *outliers*. O ano da transação foi representando com uma variável para explicar os fatores temporais no preço. Também foi adicionado ao modelo o índice de preços de habitação de Montreal o que reduziu o erro da previsão. Os algoritmos utilizados na regressão do preço foram implementados utilizando a biblioteca *scikit-learn* do *Python*. O *SVR* foi analisado com o *Kernel* linear, polinomial e gaussiano. Dentre os resultados obtidos pelo *SVR*, o melhor desempenho foi obtido pelo *Kernel* linear. No geral, o método k-NN e de

regressão de *Random Forest* tiveram melhor desempenho entre os algoritmos testados.

Plakandaras et al. (2015) propuseram uma metodologia para previsão de ameaças ao mercado imobiliário americano. Essa metodologia combina a decomposição do modo empírico do conjunto (*EEMD – Ensemble Empirical Mode Decomposition*) do campo de processamento de sinais com a metodologia de Regressão de Suporte Vetorial (*SVR – Support Vector Regression*). A capacidade de previsão desse modelo foi comparada com um modelo *Random Walk* (*RW*), um modelo autoregressivo Bayesiano (*BAR – Bayesian Autoregressive*) e um modelo autoregressivo Bayesiano de vetor (*BVAR – Bayesian Vector Autoregressive*). O modelo proposto teve desempenho superior aos demais modelos comparados. Os dados utilizados são do período de 1890 a 2012. Também foram avaliados o uso de 11 variáveis macroeconômicas. A metodologia de previsão desenvolvida também é testada como um sistema de alerta antecipado de preços, com foco na redução de preços de imóveis no mercado imobiliário americano entre 2006 e 2007, período da crise imobiliária americana. As séries temporais de previsões podem agregar elementos voláteis. As técnicas de suavização podem reduzir a influência de ruído e erros nos dados observados, oferecendo uma representação menos volátil do fenômeno. Porém, informações essenciais podem ser perdidas no processo de suavização. A implementação da metodologia explora um método de decomposição de sinal relativamente novo chamado *Ensemble Empirical Mode Decomposition* (*EEMD*). Esse método é usado como função de suavização. Após essa etapa foi utilizado Elastic Net para seleção das variáveis. Posteriormente, essas informações foram passadas para o *SVR*. O *SVR* foi utilizado com *Kernel* linear, RBF, Polinomial e Sigmóide. O conjunto de dados foi dividido em duas partes: in e out-of-sample. A proporção escolhida foi de 80/20. O melhor desempenho obtido na previsão dos preços dos imóveis foi do modelo desenvolvido com o *SVR* linear.

Mu, Wu e Zhang (2014) apresentaram um estudo de previsões de preços de imóveis no subúrbio de Boston, nos Estados Unidos, por meio de diversos métodos de aprendizagem de máquina. Os métodos são a máquina de suporte vetorial (*SVM – Support Vector Machine*), máquina de suporte de mínimos quadrados (*LSSVM – Least Squares Support Vector Machine*) e mínimo parcial quadrado (*PLS – Partial Least Squares*). Posteriormente, esses algoritmos foram comparados de acordo com os resultados previstos. Os dados foram obtidos do conjunto de dados da UCI do subúrbio de Boston que estão disponíveis na Internet. No total são 506 observações. Após a remoção de observações com dados faltantes, sobram 400 observações usadas para o treinamento e 52 observações para os testes. O melhor desempenho entre os métodos comparados foi do *SVM*. Esse método apresentou a melhor acurácia. Em seguida veio o *LSSVM* e depois o *PLS*. Pelo fato de o *LSSVM* ter uma matemática mais simplificada, esse método obteve o melhor tempo computacional. Já o algoritmo de regressão *PLS* não obteve um

bom desempenho, devido a forte presença de não linearidade nos dados utilizados.

Park e Bae (2015) desenvolveram um modelo preditivo do mercado imobiliário com algoritmos de aprendizagem de máquina. São utilizados 5.359 observações de imóveis residenciais na cidade de *Fairfax* nos Estados Unidos. Os algoritmos de aprendizagem de máquina utilizados são C4.5, *RIPPER*, *Naive Bayesian* e *AdaBoost*. Após o experimento, o desempenho desses algoritmos foi comparado. Nos Estados Unidos os preços de venda de imóveis são determinados pelo índice *Case-Shiller da Standard & Poor's* e o índice de preços da *Office of Federal Housing Enterprise Oversight* (*OFHEO*). Esses índices refletem as tendências do mercado imobiliário americano. O governo americano atua por meio de políticas governamentais para sustentar o crescimento do mercado imobiliário. Por isso esses indicadores são importantes e novos índices que representam a tendência do mercado imobiliário devem ser pesquisados e desenvolvidos. Nesse cenário de crescimento no mercado imobiliário americano, o aprendizado de máquina tem um papel importante para prever eventos futuros. A avaliação de preços de imóveis é uma decisão importante no mercado imobiliário como um todo. Inicialmente na pesquisa, foram integrados os dados de mercado imobiliário, informação de ranking de escolas públicas e a taxa de hipoteca em um único conjunto de dados. O software de mineração de dados *WEKA* foi utilizado com quatro classificadores de aprendizagem de máquina. Para determinar o desempenho de cada classificador foram realizados dois testes de desempenho. Foram extraídos 15.135 registros de três fontes de informação diferentes. Esses registros foram integrados em uma única base de dados. Cada registro possuía 76 variáveis. Das 76 variáveis, 49 foram selecionadas utilizando um teste T. Posteriormente, 28 variáveis foram selecionadas utilizando uma regressão logística gradual. Em seguida, os dados foram tratados e os valores incorretos foram removidos da base. Após a integração dos dados das três fontes, os dados foram classificados em duas classes: os que tiveram seus preços de venda acima do ofertado e os que tiveram o seu preço de venda abaixo do ofertado. A questão principal desse estudo era determinar se o preço de venda estava acima ou abaixo do preço ofertado. Para comparar os algoritmos foram explorados dois métodos de partição dos dados: divisão de três vias com 10 conjuntos ou validação cruzada com 10 conjuntos. Ambos os métodos precisam que os dados sejam divididos em 10 partes iguais e rotulados de 1 a 10. Uma dessas partes é o conjunto de testes e uma é o conjunto de validação. As demais são os conjuntos de treinamento. Para o método de divisão de três vias foram gerados 90 combinações de testes. Já para o de validação cruzada, apenas um conjunto de teste, os demais eram conjuntos de treinamento, o que fornecia 10 combinações. Dos algoritmos utilizados no trabalho, o que apresentou melhor desempenho relativo à acurácia foi o algoritmo *RIPPER* e o algoritmo *Naive Bayesian* obteve a taxa de erro mais alta.

Ng e Deisenroth (2015) apresentaram um estudo no qual analisaram e compararam diversos métodos de regressão para previsão de preços de imóveis em Londres, na Inglaterra. O Processo Gaussiano (*GP – Gaussian Process*) foi escolhido como modelo devido a sua flexibilidade e abordagem probabilística de aprendizado e escolha de modelo. O grande conjunto de dados de transações de imóveis em Londres foi tratado de forma a se ter conjuntos menores independentes por regiões. Os dados foram obtidos de órgão do governo londrino no período entre 1995 e 2013. O número total de transações de propriedades excede 2,4 milhões no período. As variáveis independentes das observações são o código da transação (ID), data da transação, preço da transação, classificação do imóvel e endereço, representado pelo código postal. O uso do Processo Gaussiano permitiu desenvolver distribuições preditivas e não apenas estimativas de pontos. Na preparação dos dados, o endereço dos imóveis representados pelo código postal foi modificado para latitudes e longitudes. Essa abordagem foi preferível por que muitas vezes vizinhos não compartilham códigos postais similares. A data da transação é representada pelo número de meses desde o período da primeira transação do conjunto de dados. As demais variáveis independentes são representadas por dígitos binários. Além disso, os preços dos imóveis foram arredondados para a casa do milhar mais próximo. O Processo Gaussiano possui uma limitação prática de 10^4 para o tamanho do conjunto de dados, pois as operações envolvidas em seus cálculos possuem complexidade O(n^3). Por isso, foi utilizada a abordagem de um Processo Gaussiano distribuído. O conjunto de dados total foi particionado por regiões de Londres. O treinamento será realizado em cada subconjunto de forma independente. Cada subconjunto deve respeitar o limite de 10.000 pontos de dados. Com isso, foram criados 2.057 subconjuntos e o total utilizado dos dados foi de 90.1%. Na realização do experimento, para cada subconjunto, foram retirados 10% dos dados para validação e os 90% restantes serviram para o conjunto de treinamento. Para a implementação do Processo Gaussiano, foi utilizada a biblioteca *GPML toolbox* do software *Matlab*. O *Kernel* utilizado no Processo Gaussiano foi o *RBF*. Além disso, foi utilizada uma função linear para média do Processo Gaussiano de forma que os preços futuros tendam para valores mais sensíveis. A abordagem de previsões desse estudo pode ser aplicada em outros problemas com variações geográficas.

Lee e Chen (2016) construíram um modelo com variáveis nítidas e difusas para prever os preços das unidades habitacionais da cidade de Taipei. Os dados foram obtidos no sítio oficial da cidade de Taipei. O foco do estudo é de imóveis usados em prédios de sete andares ou mais. Os métodos utilizados e comparados foram o *SVR* e o *ANFIS*. O trabalho recomenda que o *SVR* seja aplicado individualmente em cada distrito. O *SVR* é combinado com o algoritmo *Grid* para melhorar a combinação de parâmetros. Foram coletados 3.336 registros de transações de imóveis entre janeiro e dezembro de 2013. Imóveis muito luxuosos foram excluídos

dos dados resultando em 3.049 registros. As variáveis independentes utilizadas foram área útil, direitos na divisão da terra compartilhada individualmente, idade, número total de andares e andar do imóvel. Uma regressão gradual verifica se existe relação linear entre os produtos das variáveis e o preço. As variáveis difusas utilizadas são condição do ambiente, condição de habitação e potencial econômico. A cidade de Taipei foi dividida em 8 distritos e cada distrito foi avaliado por especialistas nas 3 variáveis difusas. As variáveis nítidas e difusas foram combinadas posteriormente. Como o número de registros em cada distrito é pequeno para treinar as redes neurais, foram selecionadas apenas 10 observações para testes e o restante foi utilizado para o treinamento. O método *SVR* obteve desempenho superior na maioria dos distritos e é o método recomendado pelo artigo.

Lim et al. (2016) apresentaram um estudo no qual investigam o conceito econométrico do modelo de preços hedônicos por meio de algoritmos de aprendizagem de máquina. Foram utilizados como algoritmos de aprendizagem o *SVR* e *k-NN* (*k-Nearest Neighbor*) e a Regressão de Comonentes Principais (*PCR – Principal Component Regression*). A abordagem de comparação de preços utilizando *PCR*, *K-NN* e *SVR* busca evitar a manipulação de preços no mercado imobiliário. Os dados de observações utilizados foram os valores de venda de imóveis entre janeiro e dezembro de 2016 da região metropolitana de Washington, nos Estados Unidos, composta por 8 cidades. Esse período corresponde ao auge da crise imobiliária americana. Esses dados foram obtidos do Serviço de Listas Múltiplas (*MLS – Multiple Listing Services*). Nesses dados foram realizados testes de distribuição normal e existiam evidências que os dados não pertenciam a uma distribuição normal. Em seguida, os dados foram normalizados e foi realizado um teste de correlação para averiguar o uso de análise de componentes principais. Além disso, foram realizados testes de *Kaiser-Meyer-Olkin* e *Bartlett* para fornecer evidências que existe relação estatística entre as variáveis. Logo, foi possível utilizar o método de decomposição de análise de componentes principais no conjunto de dados. O *PCA* foi utilizado para encontrar componentes do núcleo dos dados, aumentar a velocidade de convergência e eliminar a colinearidade dos dados. Em seguida os dados foram pré-processados, normalizados e decompostos. E após o estabelecimento da relação das variáveis, o modelo de preços hedônicos foi implementado utilizando regressão de componentes principais (PCR), regressão de suporte vetorial (*SVR*) e *k-NN*. O resultado mostrou que o desempenho do PCR leva uma pequena vantagem sobre *SVR* e *k-NN*. Além disso, o estudo validou a substituição do método de preços hedônicos por um dos algoritmos de aprendizagem de máquina utilizados.

Wu (2017) comparou diferentes métodos de seleção de características e algoritmo de extração de características, por meio de regressão de suporte vetorial, para prever o preço de imóveis na cidade de *King*, nos Estados Unidos. Os métodos de seleção utilizados foram a Eli-

minação de Características Recursivas (*RFE – Recursive Feature Elimination*), Lasso, *Ridge* e *Random Forest*. O método de extração de características utilizado foi a Análise das Componentes Principais (*PCA – Principal Component Analysis*). Após aplicar a redução de características, um modelo de regressão usando *SVR* foi construído. Para melhorar a precisão da previsão de 65% para 86% foram utilizadas as técnicas de transformação de log, redução de características e ajustes de parâmetros. O uso do *SVR* para prever preços de imóveis foi utilizado por causa das características não lineares e não estacionárias dos dados. As métricas escolhidas para avaliação são R quadrado, *MAE*, *MSE* e *RMSE*. O experimento inicial foi constituído de duas fases. Na primeira fase, inicialmente foram reduzidas as dimensões com *PCA* nas características coletadas. Em seguida, esses dados foram utilizados no *SVR*. Na segunda fase, diferentes técnicas de seleção são utilizadas para escolher as características a serem utilizadas no modelo *SVR*. Após o resultado, são calculadas as métricas de avaliação. Os dados utilizados nesse estudo são do conjunto de dados de código aberto *Kaggle*. Esses dados consistem em 20 variáveis independentes e 21.613 observações de transações de imóveis na cidade de *King*. O período registrado dessas transações é o intervalo de maio de 2014 a maio de 2015. As variáveis independentes são a data, preço, número de quartos, número de banheiros, área útil, área do terreno, andares, condição, pontuação, área acima do porão, área do porão, ano de construção, código postal, latitude, longitude, área útil em 2015, área do terreno em 2015. A variável dependente é o preço de venda. Os dados coletados são verificados para encontrar possíveis falta de valores e *outliers*. Os valores das variáveis, que são três desvios padrão afastados da média são considerados *outliers*. Os *outliers* podem ser observados visualmente por meio da biblioteca *Seaborn* em *Python*. A distribuição de preços dos imóveis se apresenta inclinada para a esquerda. Para evitar distorções foi utilizado o log do preço do imóvel. Além disso, o desempenho do modelo *SVR* também apresentou uma melhora significativa com esse ajuste. A correlação das variáveis foi analisada para evitar uma alta correlação entre as variáveis e o problema de multicolinearidade. O *PCA* foi implementado por meio da biblioteca Scikit-learn em Python. Essa biblioteca utiliza decomposição singular de valor (*SVD – Singular Value Decomposition*) para implementar a redução da dimensão de *PCA*. São escolhidos 16 principais componentes de análise baseados na menor taxa de erro. A partir disso, o método *RFE* constrói um ranking das variáveis utilizando *SVR* com *Kernel* linear e modelo linear regular como modelo estimador. Já os métodos Lasso e *Ridge* utilizam o conceito de coeficientes do modelo de regressão para criar um ranking de importância das variáveis. Eles podem ser considerados modelos de regularização. O *Random Forest* é um algoritmo construído com árvores de decisão, no qual cada nó é uma condição da variável. A importância das variáveis será calculada por meio da biblioteca *Sklearn*. O *SVR* é implementado com a biblioteca *Scikit-learn* em *Python*.

Serão comparados resultados com diferentes *Kernel* e valores de parâmetros C e épsilon. O *Kernel* que acompanhou melhor a tendência dos dados foi o *Kernel RBF*. Os resultados do *SVR* sem redução de variáveis que apresentou melhor resultado utilizou um *Kernel* linear com o valor de C=1000. Quando foi utilizada a redução de variáveis por meio do *PCA* o *SVR* que apresentou melhor resultado também utilizou o *Kernel* linear. Em seguida, o *SVR* utilizou diversas técnicas de seleção de variáveis. Os resultados mostraram que não há diferenças entre o desempenho das seleções de variáveis e a extração de variáveis. O melhor resultado geral foi obtido utilizando *SVR* com *Kernel RBF* e C=10.

Trawiński et al. (2017) apresentaram um estudo no qual são comparados algoritmos especialistas com modelos de aprendizagem de máquina para avaliação imobiliária. Os algoritmos especialistas poderiam ser utilizados quando o tamanho do conjunto de dados não fosse suficiente para os modelos de aprendizagem de máquina. No estudo foram apresentados dois algoritmos especialistas para auxiliar a avaliação de imóveis. Ambos os algoritmos utilizam a abordagem de comparação de vendas. O primeiro algoritmo utiliza um número de transações anteriores de imóveis similares em uma determinada região do mercado. O segundo algoritmo calcula o preço estimado como uma média de valores de um número de imóveis próximos que pertencem a uma mesma classe de imóvel. O desempenho dos dois algoritmos foi testado com a utilização de dados reais de transações imobiliárias. Posteriormente, esses algoritmos foram comparados com três modelos de regressão criados no software *WEKA*. Os dados de imóveis se referem a um município polonês. Esses dados foram obtidos em um registro público de transação imobiliária desse município. As variáveis chaves utilizadas foram área útil, ano de construção do prédio, número de andares no prédio e número de quartos incluindo a cozinha. Após o tratamento dos dados, sobraram 12.439 registros de vendas de imóveis no período de 1998 a 2013. Todos os preços de imóveis foram atualizados para o último dia do período analisado com base em uma tendência de mercado de mudança de preço de propriedade. O conjunto de dados foi dividido na proporção de 30% para os testes e 70% para o treinamento dos algoritmos de aprendizagem de máquina. O conjunto de treinamentos foi dividido em 69 subconjuntos. Cada subconjunto representava uma região cadastral individual. Em cada região havia pelos menos 50 registros de transações de imóveis. Para cada região foi utilizada três tipos de modelos de previsão nos dados de treinamento. Esses modelos foram utilizados no software *WEKA*: *M5P* (*Pruned Model Tree*), *MLR* (*Multilayer Perceptron*) e *LR* (*Linear Regression*). Cada algoritmo utilizou quatro variáveis de entrada (variáveis independentes) e a saída era o preço por metro quadrado (variável dependente). Como resultado do trabalho, os algoritmos de aprendizagem de máquina tiveram um desempenho superior aos algoritmos especialistas. Entre os algoritmos de aprendizagem de máquina o *M5P* obteve desempenho superior em relação aos demais.

2.3.3 Redes Neurais Artificiais

Limsombunchai (2004) realizou uma comparação empírica entre o modelo hedônico e as redes neurais artificiais na previsão de preços no mercado imobiliário de uma cidade da Nova Zelândia. Um conjunto de dados de 200 casas em *Christchurch*, na Nova Zelândia, foi selecionado de forma aleatória de um sítio na Internet de uma corretora de imóveis. As seguintes características das casas foram analisadas: tamanho, idade, tipo, número de quartos, número de banheiros, vagas na garagem, amenidades no entorno e localização geográfica. A escolha da forma funcional para o modelo hedônico foi um desafio nesse trabalho. Nesse trabalho optou-se por utilizar o modelo de semi-log porque o preço é um componente sensível e volátil. As redes neurais tiveram um desempenho superior ao modelo hedônico de preços. A falta de atributos ambientais, o número inadequado de dados e a relação não linear entre os atributos do imóvel e seu preço podem afetar o modelo hedônico de preços. Porém a rede neural foi superior apenas ao se utilizar o método de tentativa e erro. Além disso, o estudo apresentou algumas dificuldades, já que o preço utilizado não é o preço real de venda e não foi considerado o componente de tempo no preço do imóvel. Esse preço também poderia sofrer variações de fatores econômicos que não foram incluídas no modelo.

Guedes (1999) comparou os resultados de avaliação de preços de imóveis comerciais no centro de Buenos Aires, na Argentina. As estimativas dos preços foram obtidas por meio de uma análise de regressão linear e redes neurais artificiais. Os dados utilizados foram de um prédio comercial no centro de Buenos Aires. A regressão linear permite analisar as influências dos atributos na variável independente. É possível fazer simulações ao modificar alguns parâmetros e manter outros constantes. Já as redes neurais são matematicamente mais sofisticadas e são mais difíceis de quantificar de imediato as influências dos atributos dos imóveis. A base de dados possui 48 observações. Na análise de regressão, o erro quadrático médio (EQM) foi de 45,404. Na técnica de redes neurais o EQM foi de 30,217. Nesse estudo foi constatado que o modelo de redes neurais obteve melhores resultados do que a análise de regressão. O EQM obtido por meio das redes neurais foi mais de 30% menor do que aquele gerado pela análise de regressão múltipla.

Lin e Chen (2010) apresentaram um estudo para previsão de preço de imóveis em Taiwan, na China, utilizando redes neurais artificiais e regressão de suporte vetorial (*SVR – Support Vector Regression*). Inicialmente as variáveis foram escolhidas de pesquisas anteriores e por meio de um procedimento de tentativa e erro. O método de regressão de suporte vetorial apresentou o melhor desempenho em comparação com as redes neurais com *Back Propagation* (*BPNN – Back Propagation Neural Network*). Os dados foram coletados do Portal Imobiliário de Taiwan.

Após a coleta dos dados, foram calculadas as médias mensais dos preços dos imóveis. A *BPNN* utilizou a técnica de *Feedforward* com *Backward Error Propagation*. Isso significa que a rede neural aprende com exemplos. As vantagens desse tipo de rede neural incluem a habilidade de tratar dados não lineares, a tolerância a erros, permitir a computação paralela e de ser um aproximador universal. Essas vantagens só estão presentes caso existam dados suficientes. O modelo *SVR* depende de um subconjunto de treinamento dos dados. Esse modelo também pode tratar dados não lineares e possui apenas uma solução ótima a partir de um conjunto de parâmetros do Kernel. O *SVR* é combinado com algoritmo de Grid para melhorar a combinação de parâmetros. Muitas previsões dos preços dos imóveis realizadas pelo modelo SVR foram melhores do que a das redes neurais. Além disso, foram utilizadas as seguintes variáveis tanto para BPNN quanto para SVR: taxa de redesconto, fornecimento de dinheiro e preço do mês anterior. Para o estudo, essas foram as variáveis mais importantes para a previsão de preços imobiliários. Além dessas variáveis, para obter uma melhor previsão usando SVR foi utilizado alguns indicadores econômicos como, por exemplo, o Produto Interno Bruto.

Teixeira e Ocerín (2011) apresentaram um estudo para previsão dos preços de vendas de apartamentos em uma cidade Portuguesa de Castelo Branco por meio de redes neurais artificiais. Nesse estudo foram realizados três partições dos dados. A partição com melhor desempenho foi a que apresentava a proporção de 90% dos dados para o conjunto de treinamento e 10% dos dados para o conjunto de teste. Outro aspecto analisado pelo estudo foi a importância das variáveis explicativas do preço. Nesse caso, a variável de área útil foi a que teve maior importância. A base de dados foi obtida de agentes imobiliários da cidade analisada. Essa base possuía 200 registros de apartamentos vendidos na cidade de Castelo Branco entre 2005 e 2009. Foram criados 5 índices para agrupar as características qualitativas da amostra. Os índices foram os seguintes: conforto, anexos, conservação, interno e externo. Além desses índices, foi criado um índice de localização. Esses índices variam no intervalo de 0 a 1. Isso ocorre para que eles sejam mais homogêneos possíveis e que tenham a mesma importância relativa entre eles. A estimação do preço de imóveis em Castelo Branco foi realizada por meio do software estatístico *SPSS*, v.18. Vários testes foram realizados com diferentes conjuntos de dados de treinamento e testes gerados aleatoriamente pelo software *SPSS*. A rede neural construída possuía três camadas, com sete neurônios na camada de entrada e quatro neurônios na camada oculta e apenas um neurônio na camada de saída. A rede neural obteve um erro relativo de apenas 0,063. Logo, o estudo apontou que a rede neural possuía uma eficiência de 93,7%.

Bahia (2013) realizou um estudo para prever os preços de imóveis no mercado imobiliário utilizando mineração de dados e redes neurais. Nessa pesquisa, são utilizados os modelos

de redes neurais artificiais *FFBP* (*Feed Forward Back Propagation Artificial Neural Network*) e *CFBP* (*Cascade Forward Back Propagation Neural Network*). Os dados foram obtidos do conjunto de dados do repositório de aprendizagem de máquina da UCI (*UCI Machine Learning Repository*). O conjunto de dados se refere aos imóveis no subúrbio de Boston nos Estados Unidos. Este conjunto de dados possui 506 amostras e 13 variáveis. A camada de entrada é uma matriz de 13 x 506 e a camada de saída é uma matriz de 1 x 506 de valor médio de imóveis. A métrica utilizada na avaliação é o *MSE* (*Mean Square Error*). Os dados são divididos em 80% para treinamento e 20% para testes. O melhor resultado obtido foi pela Rede *CFBP*.

Khamis e Kamarudin (2014) compararam o desempenho da previsão de preços de imóveis na Cidade de Nova York, nos Estados Unidos, entre os métodos de Regressão Linear Múltipla e Redes Neurais Artificiais. Os dados observados são uma amostra aleatória de 1.047 imóveis obtidas do sítio *Math10*. As variáveis independentes escolhidas foram área útil, número de quartos, número de banheiros, tamanho do terreno e idade. A variável dependente escolhida foi o preço do imóvel. Os resultados mostraram que o desempenho da rede neural é superior ao da regressão linear múltipla.

Lin e Chen (2015) discutem a tendência do preço unitário médio de imóveis na cidade de Taipei, capital de Taiwan. Com isso, o estudo espera estabelecer um modelo de previsão de preços e os fatores-chave que determinam esse preço. Nesse estudo, são comparados métodos estatísticos tradicionais, Redes Neurais e Regressão por Suporte Vetorial. Os resultados desse estudo mostraram que o *SVR* superou as Redes Neurais e o procedimento passo a passo é válido na seleção de variáveis. Além disso, o preço de negociação anterior (preço em t-1), fornecimento de dinheiro e empréstimos para casas novas são os fatores-chave na determinação do preço do imóvel. Os dados observados foram obtidos do portal imobiliário de Taiwan. Após a coleta dos dados, foram calculadas as médias mensais. Os dados foram coletados entre os períodos de 2004/1 e 2008/12. Para realização do experimento, a rede neural com *Backward Error Propagation* e *Feedforward* com 3 camadas foi escolhida. Essa rede neural tem a vantagem de aprender com exemplos. Já o *SVR* foi combinado com o algoritmo *Grid* para melhorar a combinação dos parâmetros. Além disso, para melhorar a eficiência do modelo, 12 variáveis de indicadores imobiliários e econômicos são utilizadas. Para o treinamento das redes neurais foi utilizado a biblioteca *NN* no *Matlab 2008*. Para o *SVR* foi utilizado o *Kernel RFB* e o algoritmo *Grid* é utilizado para melhorar a combinação dos parâmetros do *Kernel*. A seleção de variáveis foi realizada por um procedimento gradual e por tentativa e erro. O estudo sugere que o preço do imóvel em Taipei é mais relacionado com financiamento e investimentos existentes do que as características do próprio imóvel.

Lim et al. (2016) mostraram o desempenho das previsões de preços de imóveis em Sigapura por meio de uma rede neural artificial com multicamadas Perceptron, com um modelo autoregressivo integrado de médias móveis (*ARIMA – Autoregressive Integrated Moving Average*). Esse modelo é comparado com um modelo de análise de regressão múltipla (*MRA – Multiple Regression Analysis*). O modelo com melhor desempenho foi utilizado para prever o índice de preços imobiliários futuros (*CPI – Condominium Price Index*). As variáveis independentes consideradas no modelo de rede neural são gasto do consumidor, média salarial, produto interno bruto, índice de preços do consumidor, taxa de empréstimos, taxa de juros, população, o índice de preços *Singapore Housing and Development Board* (*HDB*), mudanças no índice *HDB*, índice de tempo *Straits* e número de unidades disponíveis. A variável dependente é o índice de preços *CPI*. Os dados das séries temporais foram obtidos no sistema *REALIS* do sítio do Departamento de Estatística e Economia de Singapura (*Singapore Department of Satistics and Trading Economy*). Os dados foram coletados do período de 1990 a 2013. A maioria dos dados foi coletada trimestralmente. Os dados disponíveis anualmente foram convertidos para trimestrais por meio de uma interpolação de *spline* cúbica (*CSI – Cubic Spline Interpolation*). A rede neural possuía apenas uma camada oculta com 10 neurônios. A rede neural deve prever 92 valores dos índices começando em 1991 (T1) até 2013 (T4). As redes neurais artificias provaram ter um desempenho superior à análise de regressão múltipla na previsão do índice de preço. Porém o estudo indica a falta de dados para treinamento com um problema.

3 METODOLOGIA

A partir do estudo bibliográfico foi possível compreender que os estudos em relação a avaliação, precificação e previsão de preços de imóveis utilizaram, inicialmente, métodos tradicionais de regressão linear, e com o tempo, passaram a utilizar métodos mais complexos e sofisticados, que utilizam redes neurais artificiais. O uso de redes neurais utilizados nesses estudos permitiu obter melhores resultados que os métodos tradicionais de regressão.

3.1 Regressão Linear

O método de regressão linear múltipla utilizada em vários trabalhos analisados no referencial teórico pode ser expresso por:

$$Y = Xw + \beta_0 + \varepsilon \tag{3.1}$$

Na qual a variável dependente é o vetor Y, a variável independente é o vetor X. E o w é vetor de parâmetros. O β_0 representa o intercepto e o ε é o erro aleatório.

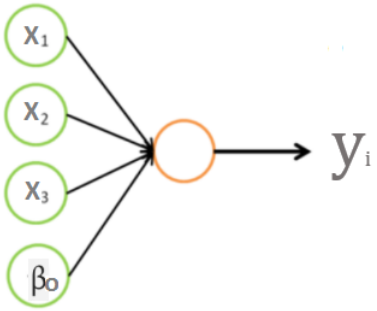


Figura 3.1 – Regressão Linear

Fonte: Elaborado pelo autor.

A Figura 3.1 mostra uma representação de uma regressão linear com três variáveis independentes X_i, o intercepto β_0 e a variável dependente y_i.

A regressão linear é representada por um modelo matemático que descreve o relacionamento entre duas ou mais variáveis, com a finalidade de estimar valores para uma variável (dependente), com base em valores conhecido de outras (independentes). No caso de apenas uma variável independente é chamado de regressão linear simples. Para mais de uma variável independente, o processo é chamado de regressão linear múltipla.

A regressão linear pode ser vista como uma rede neural com um único neurônio, se representada dessa forma:

$$Y = Xw + \varepsilon \tag{3.2}$$

Na qual a variável dependente é o vetor Y, a variável independente é o vetor X. E o w é vetor de parâmetros. E ε é o erro aleatório.

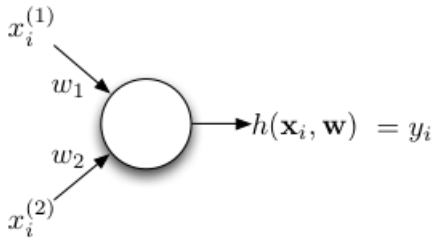


Figura 3.2 – Regressão linear como caso específico de uma rede neural

Fonte: Adaptado de Dolhansky (2013).

A Figura 3.2 mostra uma regressão linear como um caso específico de uma rede neural com duas variáveis $X_i^{(1)}$ e $X_i^{(2)}$. Além disso, w_1 e w_2 são os parâmetros e a variável dependente é y_i.

3.2 Redes Neurais Artificiais

A evolução das regressões lineares com o complemento de não-linearidade, pode-se utilizar as Redes Neurais Artificiais (RNA). A Rede Neural Artificial é uma das técnicas mais recentes. São modelos matemáticos inspirados no funcionamento de células neurais, ou seja,

uma estrutura de neurônios inteligentes, que adquirem conhecimento por meio da experiência (treinamento) e podem resolver problemas.

Uma rede neural é formada por um conjunto de neurônios ou unidades de processamento. Essas unidades de processamento são distribuídas em um conjunto de camadas. As redes neurais que apresentam uma ou mais camadas de entrada e saída são chamadas de Redes Neurais Perceptron de Múltiplas Camadas (*MLP- Multilayer Perceptron*).

Além disso, nas redes neurais as arquiteturas podem ser rasas ou profundas, dependendo do número de camadas ocultas.

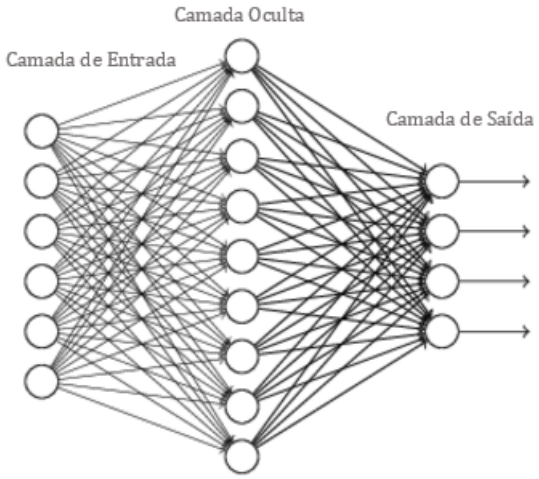


Figura 3.3 – Rede Neural (Arquitetura Rasa)

Fonte: Adaptado de Nielsen (2018a).

A Figura 3.3 mostra uma rede neural de arquitetura rasa.

3.3 Redes Neurais Rasas

Na Rede Neural com arquitetura rasa pode ser representada pela equação 3.4.

$$Y = \psi'(XW)w_0 \tag{3.3}$$

Onde a variável dependente é o vetor Y, a função de ativação é ψ, a variável independente (dados) é o vetor X. W representa os pesos da camada da rede e por último, w_0 é o peso da camada de saída.

A equação acima, com a função de ativação como uma função identidade, é igual a um modelo de regressão linear, ou seja, uma rede neural retirada a não linearidade está sujeita às mesmas restrições que os modelos lineares.

3.4 Redes Neurais Profundas

A Rede Neural que utiliza uma arquitetura profunda é representada pela equação:

$$Y = \psi_1(\psi_2(\psi_3 \cdots (X_1 W_1) X_2 W_2) X_3 W_3) \cdots W_l X_l) w_o \tag{3.4}$$

Onde ψ são funções de ativação e o número l representa o número de camadas ocultas. O vetor de dados X_i e W_i são os pesos das camadas ocultas O w_o representa os pesos da camada de saída.

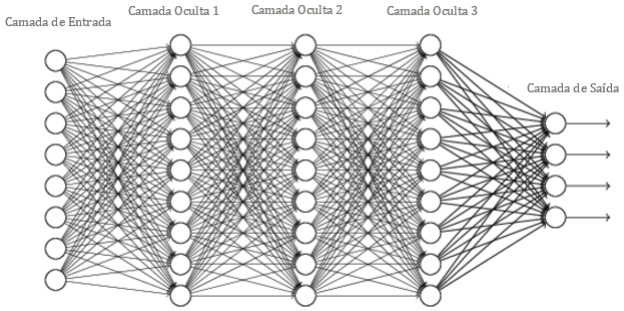


Figura 3.4 – Rede Neural (Arquitetura Profunda)

Fonte: Adaptado de Nielsen (2018b).

A Figura 3.3 mostra uma Rede Neural de Arquitetura Profunda.

Redes neurais profundas podem ser consideradas uma evolução em relação às redes neurais de arquitetura rasa, na medida em que são capazes de aprender a mesma informação de uma rede neural de arquitetura rasa com menos parâmetros (GOODFELLOW; BENGIO; COURVILLE, 2016).

3.5 Redes Neurais Recorrentes

No âmbito das redes neurais existem as redes neurais recorrentes (RNR). A camada oculta da rede neural recorrente recebe informações tanto da camada de entrada quanto da camada

oculta na iteração de tempo anterior. Esse tipo de rede neural funciona como se tivesse memória, ou seja, a cada período de tempo, a rede neural armazena as informações do seu estado oculto naquele período de tempo, como também recebe informações do estado oculto anterior, que armazenou toda a informação relevante que aconteceu no passado. Além disso, esse estado oculto nesse período pode fornecer informações para a camada de saída, caso seja o momento de realizar a previsão do modelo e, em seguida, transmite informações para o estado oculto do próximo período de tempo.

A rede neural recorrente pode ser considerada uma rede neural de arquitetura profunda ao longo do tempo. Em cada período de treinamento, as camadas ocultas da RNR armazenam informações do respectivo período e informações relevantes do passado vão sendo repassados em diante (GOODFELLOW; BENGIO; COURVILLE, 2016). Logo, os dados utilizados na rede neural são variáveis defasadas no tempo. Porém, deve-se decidir quantas defasagens é suficiente para realizar uma previsão confiável. Outra dificuldade é o número de variáveis em determinado período e possível ausência em determinados períodos defasados.

A rede neural recorrente pode processar sequências independentemente do seu tamanho e das suas variações e pode ser representada pela seguintes equações:

$$\begin{aligned} h_t &= \psi_1(b_h + xW_x + h_{t-1}W_h) \\ \hat{y}_{t+1} &= b_0 + h_t W_0 \end{aligned} \tag{3.5}$$

Recebem informações das variáveis independentes X_t e processam esses valores resultado no processamento h_t. Na equação acima é possível ver que o estado oculto do período anterior $h_{t-1}W_h$ é adicionado ao modelo da rede neural clássica. Assim, essas redes neurais compartilham parâmetros ao longo do tempo. E o vetor W_0 representa os pesos da camada de saída. O período de tempo utilizado pelo modelo de redes neurais recorrentes representa uma ordem em uma sequência e não o tempo na sua concepção mais conhecida.

Uma rede neural recorrente com uma camada oculta, sendo processada por 2 períodos de tempo e realizando a previsão de uma variável contínua pode ser expressa pelas seguintes equações:

$$H_0 = \psi(b_h + XW_x) \tag{3.6}$$

$$H_1 = \psi(b_h + XW_x + H_0W_h) \tag{3.7}$$

$$\hat{y} = b_0 + H_2W_0 \tag{3.8}$$

$$\mathscr{L} = \frac{1}{n}\Sigma_{i=0}^{n}(\hat{y} - y)^2 \tag{3.9}$$

No trabalho em tela será utilizado uma aplicação de rede neural recorrente. Nesse caso, tempos uma sequência na entrada e na saída da rede. Será realizado uma previsão para o período de tempo seguinte, dado o que ocorreu no período de tempo atual e nos períodos anteriores. A Figura 3.5 mostra uma Rede Neural Recorrente.

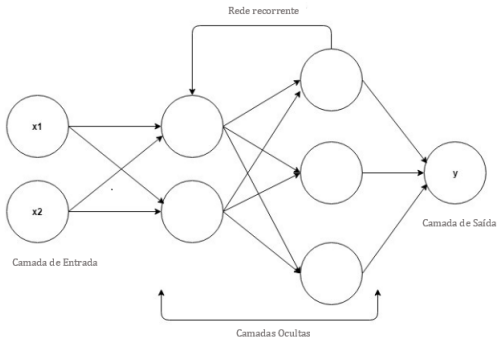


Figura 3.5 – Rede Neural Recorrente

Fonte: Adaptado de Santos (2018).

3.6 Redes Neurais Recorrentes *LSTM*

A pesquisa utiliza o modelo de aprendizado de máquinas "*Long Short-Term Memory*"(*LSTM*). Esse modelo de aprendizado é baseado em redes neurais recorrentes. As redes neurais recorrentes podem ser utilizadas para previsão da série temporal de preços de imóveis no Distrito Federal. Essas redes tem a vantagem de aprender e lembrar-se de sequências longas. As *LSTMs* são um tipo especial de RNR que possuem *loops*. Esses *loops* permitem que as informações persistam, assim as informações são passadas de uma etapa da rede para a seguinte. Porém, essas redes também consideram o descarte de informações irrelevantes. Todas as redes neurais recorrentes têm a forma de uma cadeia de módulos repetitivos da rede neural. Essa rede pode ser imaginada como múltiplas cópias da mesma rede, cada uma passando a mensagem a um sucessor conforme Figura 3.6.

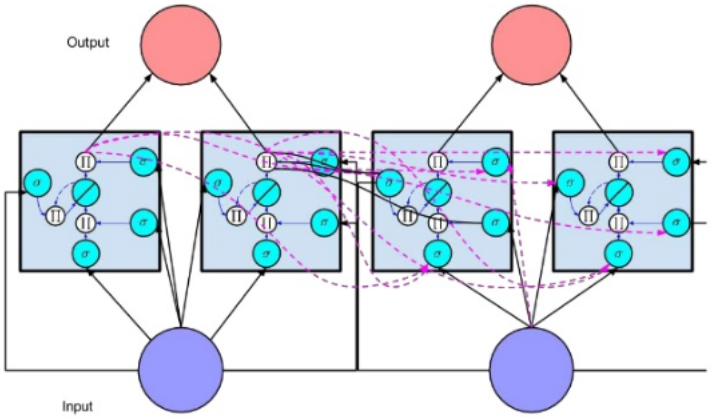


Figura 3.6 – Rede Neural LSTM

Fonte: Lipton (2015).

A estrutura encadeada das redes neurais se relaciona de forma complementar com sequências e listas. Essas estruturas são a arquitetura natural da rede neural, que pode ser facilmente adaptada para lidar com séries de dados, ou no caso do trabalho realizado, de séries temporais. Um dos benefícios das RNRs é que elas podem conectar informações anteriores à tarefa atual. Por isso, essas redes são bastante úteis para a previsão de série temporais. Além de lidar com dependências de memória longa, essas redes são capazes de lidar com o problema do *"vanish gradient"*.

As redes neurais recorrentes por permitirem a modelagem de tarefas de dados sequenciais longas sofrem do problema do *"vanish gradient"*. Esse problema é ocasionado ao se transportar informações usadas na atualização da RNR quando o gradiente se torna pequeno e as atualizações tornam-se insignificantes. Isso é ocasionado por longas séries de multiplicações de pequenos valores, o que diminui o valor dos gradientes e faz com que não ocorra nenhum aprendizado real. As redes *LSTM* resolvem o problema do desaparecimento dos gradientes criando uma conexão entre as portas da célula que criam um caminho para o fluxo de informações que não podem ser esquecidas.

A biblioteca utilizada para implementar a RNR *LSTM* é a *Keras*. Para poder se lembrar de sequências longas, essa biblioteca pode ser utilizada com o argumento *"stateful"* como verdadeiro ao definir a camada da rede *LSTM*. Essa camada mantém o estado entre os dados de um lote. Um lote de dados é um número fixo de linhas do conjunto de dados de treinamento que define quantos padrões devem ser processados antes de atualizar os pesos da rede. No modo *"stateless"*, a biblioteca *keras* embaralha as amostras e as dependências entre a série temporal e a versão atrasada de si são perdidas. Quando o *keras* é executado no modo *"stateful"* é pos-

sível obter resultados com alta precisão, pois o último estado para cada amostra no índice *i* no lote, será usado o estado inicial para a amostra do índice *i* no lote seguinte.

3.6.1 Especificação da Rede

O modelo *LSTM* assume que seus dados são divididos em componentes de entrada (X) e saída (y). Para um problema de série temporal, podemos fazer isso usando a observação da última etapa (t-1) como entrada, e a etapa de tempo atual (t) como a observação da saída para enquadrar uma sequência como um problema de aprendizado supervisionado. Cada etapa de tempo na sequência original é separada em uma etapa de tempo (*timestep*) e uma característica (preço). Os dados de entrada devem ser especificados na camada *LSTM* usando o argumento *"batch_input_shape"* como uma tupla que especifica o número esperado de observações ou amostras para ler cada lote, o número de etapas de tempo e o número de características.

Será utilizado uma única etapa de tempo (Modelo de um passo a frente) e 12 etapas de tempo (Modelo de janela móvel). Para o modelo de um passo a frente a matriz será no formato (179, 2) E para o de janela móvel no formato (179, 13).

- Amostra: são as observações independentes do domínio. O valor da amostra é de 180 observações, referente ao período de janeiro de 1997 até dezembro de 2011. Essa amostra será dividida em um período de treinamento (120) e testes (60).
- Número de treinamento: representa um subconjunto das amostras que servirá para treinar a rede neural. O valor desse parâmetro é de 120, que compreende o período de janeiro de 1997 a dezembro de 2006.
- Número de testes: representa um subconjunto das amostras que servirá para testar a previsão da rede neural. O valor desse parâmetro é 60, que compreende o período de janeiro de 2007 a dezembro de 2011.
- Número de novas previsões: representa o número de previsões realizadas após os valores observados na amostra, que compreende o período de janeiro de 2012 a dezembro de 2012 (12 meses).
- Épocas (*epochs*): são o número total de iterações de avanço/retrocesso. É uma variável que tende a melhorar o desempenho do modelo, a menos que o *overfitting* ocorra. O *overfitting* ocorre quando um modelo aprende os detalhes e o ruído nos dados de treinamento e isso afeta o desempenho do modelo em novos dados. Seu valor será de 200.

- Etapas do tempo (*steps*): são as etapas de tempo separadas de uma dada variável para uma dada observação. É o número de defasagens incluídas no conjunto de treinamento/teste. Esse valor será de 1 para o modelo de um passo a frente e de 12 para o modelo de janela móvel.

- Tamanho do lote (*batchsize*): o tamanho do lote é geralmente muito menor que o número total de amostras. O tamanho do lote, juntamente com o número de épocas (*epochs*), definem a rapidez com que a rede aprende os dados e com que frequência os pesos são atualizados. Esse valor será de 1.

- Características (*features*): Estas são medidas separadas observadas no momento da observação. Apenas o preço do imóvel é utilizado, logo esse parâmetro é 1.

Por último é definido o número de neurônios, ou número de unidades de memória. No caso em tela, optou-se por um número de 1 neurônio. Além disso, a rede utiliza uma única camada oculta do *LSTM* que também especifica as expectativas da camada de entrada através do argumento "*batch_input_shape*". A rede possui um único neurônio na camada de saída com uma ativação linear para realização da previsão do preço do imóvel no próximo período de tempo.

Após a especificação da rede, ela será compilada na biblioteca *TensorFlow*. Na compilação da rede, também é especificado uma função de perda e um algoritmo de otimização. A função de perda ou *loss* será a "*mean_squared_error*"que corresponde ao *RMSE* (*Root Mean Square Error*), e o algoritmo de otimização será o ADAM.

A medida de erro *RMSE* ou em português raiz do erro médio quadrático pode ser utilizado como métrica para se avaliar a qualidade do modelo utilizado e, em consequência, a qualidade da previsão. A raiz quadrada do valor médio quadrático mede o grau de dispersão em torno da linha de regressão e é dado pela seguinte equação:

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(d_i - y_i\right)^2} \tag{3.10}$$

Onde n é o número de entradas, d_i são os valores do modelo e y_i são os valores observados.

Essa será a medida utilizada para avaliar os modelos propostos.

3.7 Preparação dos dados e Avaliação do Modelo

No desenvolvimento do trabalho, foram utilizados dados reais sobre preços de imóveis na região do Distrito Federal. Esta seção descreve a preparação dos dados e a avaliação dos modelos usados nesse trabalho.

3.7.1 Coletar os dados

A pesquisa realizada utilizou dados da Caixa Econômica Federal de imóveis na região do Distrito Federal para o período de 1 de janeiro de 1997 até 31 de dezembro de 2011. Esse período compreende a utilização da mesma moeda no Brasil, o Real, o que facilita a utilização dos dados e a comparação das informações. Além disso, o período utilizado abarca mais de um ciclo econômico, com o objetivo de compreender melhor o comportamento dos preços em cada região independente de ciclos econômicos favoráveis ou desfavoráveis.

As séries temporais são compostas do CEP e a média dos preços dos imóveis no mês. Por isso, elas foram divididas em 26 CEPs em várias subsequências menores. Logo, as *LSTMs* processam para cada CEP uma amostra em que cada amostra é uma única série temporal.

3.7.2 Remover a coluna de data

A série temporal utilizada possui valores uniformes ao longo do tempo e não possui valores omissos. A coluna de tempo foi removida. Quando os valores omissos estão presentes, são utilizados o último valor disponível.

3.7.3 Deflacionar os preços pelo índice correspondente

Após a escolha do período foi necessário realizar um ajuste nos dados. O principal ajuste considerado foi deflacionar a série de preços de imóveis com um índice representativo da inflação nesse setor. O índice de inflação escolhido que melhor representa a variação dos preços dos imóveis em decorrência da inflação, é o IGPDI - Índice Geral de Preços - Disponibilidade Interna. Esse é o índice utilizado no reajuste de preços de aluguéis.

3.7.4 Remoção dos *outliers*

O segundo passo no ajuste dos dados foi a remoção dos *outliers* para não contaminar os dados. Em alguns casos, esses dados representam algum erro na base de dados e podem ser desconsiderados. No lugar dos *outliers* foi repetido o último valor válido da série temporal. Para identificar quais valores são *outliers* foi calculado o desvio padrão dos dados, e valores que são superiores ou inferiores em até três vezes o desvio padrão em relação a média da amostra, foram substituídos.

3.7.5 Transformando a série temporal em estacionária

O preços de imóveis possuem uma estrutura nos dados que dependem do tempo, ou seja, há uma tendência de os preços crescerem com o passar do tempo. Logo, a série temporal não é estacionária. Os dados estacionários nos fornecem previsões mais precisas e por esse motivo a série temporal utilizada foi modificada para que a tendência fosse removida. Uma observação em um passo de tempo anterior (t-1) foi subtraída da observação atual (t). Essa operação foi realizada por meio de uma função em *Python* que calcula a diferença. Isso resulta em uma série temporal de diferenças, ou de mudanças de observações de um passo para outro.

Após a aplicação e obtenção das previsões, deve-se inverter o procedimento para obter a escala original. Além disso, o valor original será utilizado para calcular o erro do modelo em relação ao realizado.

3.7.6 Transformando os dados da série temporal para uma escala específica

O terceiro passo no ajuste dos dados foi a normalização da série temporal em valores entre -1 e 1. Essa operação foi realizada por meio da classe em *Python MinMaxScaler* da biblioteca *sklearn*. As redes neurais *LSTMs* esperam que os dados estejam dentro da escala da função de ativação usada pela rede.

$$\begin{aligned} X_{std} &= \frac{X - X_min}{X_max - X_min} \\ X_{scaled} &= X_{std} * (max - min) + min \end{aligned} \tag{3.11}$$

Onde max é 1, min é igual a -1. E X_{max} e X_{min}, os valores de máximo e mínimo.

A função de tangente hiperbólica (tanh) é a função de ativação padrão das redes *LSTMs*. Essa função produz os valores entre -1 e 1 conforme Figura 3.7. E o valor da derivada atinge o máximo de 1 quando $x = 0$. Por esse motivo recomenda-se o uso dessas funções de ativação no lugar da sigmoide.

A saída da função tanh pode ser positiva ou negativa, o que permite aumentos ou diminuições no estado. Esse é um dos motivos desse tipo de função ser utilizado para determinar os valores candidatos para ser adicionado ao estado interno.

Logo, os valores dos coeficientes de escala mínimo e máximo devem ser calculados no conjunto de dados de treinamento e aplicados para dimensionar o conjunto de dados de teste e as previsões. Isso tem como objetivo evitar a contaminação do experimento com o conhecimento do conjunto de dados de teste, o que pode fornecer ao modelo uma pequena vantagem.

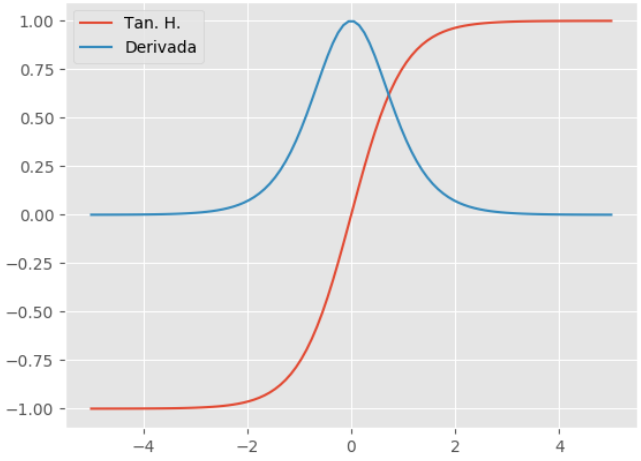


Figura 3.7 – Função de ativação Tangente Hiperbólica

Fonte: Facure (2017)

Após a aplicação e obtenção das previsões, a escala é invertida para retomar os valores à escala original, para que os resultados possam ser interpretados corretamente. Além disso, a escala original é utilizada para dimensionamento do erro do modelo.

Será utilizada a média mensal dos preços dos imóveis e essa série temporal será transformada em um problema de aprendizado supervisionado. O modelo será desenvolvido usando o conjuntos de dados de treinamento e serão feitas previsões no conjunto de dados de teste.

3.8 Desenvolvimento do Modelo *LSTM*

O primeiro passo para realizar as previsões com o uso das redes *LSTMs* foi a criação de um *baseline* de previsão para comparação com os métodos utilizados.

Para a série temporal de preços no trabalho com uma tendência crescente linear foi utilizado um *baseline Random Walk*, ou passeio aleatório. Esse modelo de previsão utiliza a observação da etapa anterior X_{t-1} para prever a observação na etapa de tempo atual X_t conforme mostrada na equação (CHATFIELD, 2016).

$$X_t = X_{t-1} + Z_t \tag{3.12}$$

Onde Z_t é um processo discreto, puramente aleatório, com média de μ e variância σ_z^2.

A partir de um *baseline* é possível desenvolver e avaliar as redes neurais recorrentes *LSTMs* para a previsão de séries temporais dos dados do trabalho.

No modelo proposto, a previsão *LSTM* utiliza uma série temporal com apenas uma variável a ser analisada. Além disso, é utilizado o método de um passo a frente (*One step method*) e de janela móvel (*Window Method*). No método de um passo a frente, a última etapa define a previsão. Já no método de janela móvel, são utilizados vários valores anteriores para determinar a previsão.

Após a realização das previsões, o passo final é avaliar as previsões, comparar os resultados e analisar qual método apresentou o melhor resultado. A avaliação das previsões pode ser realizada por meio do cálculo do *RMSE* dos modelos propostos.

3.9 Tecnologia Utilizada

Inicialmente, para tratamento dos dados, foi utilizada planilhas *Microsoft Excel* Versão 2016. Posteriormente, foi utilizado o ambiente de desenvolvimento *SPYDER* (*Scientific PYthon Development EnviRonment*) versão 3.3.3 para a linguagem *Python* versão 3.7. O *SPYDER* contém um editor especializado, integrado em uma interface gráfica em que se pode visualizar saídas numéricas e gráficas. Além do console *Python* interativo, existe um visualizador de documentação. A distribuição utilizada é a Anaconda *Navigator* versão 1.9.7, e nela está contido o ambiente de desenvolvimento *SPYDER*. Antes de executar o programa é necessário instalar as bibliotecas *Keras* e *Tensorflow*. Para apresentação do mapa de calor foi utilizado a ferramenta

Qliksense Desktop 13.21.1.

4 RESULTADOS

Nessa seção serão apresentados os resultados obtidos com o uso das redes neurais recorrentes conforme descrito na seção de metodologia. Os resultados serão demonstrados por meio de tabelas e apresentado aqui para um único CEP de forma a exemplificar o processo realizado. Ao final serão consolidados as previsões para todos os CEP em um único mapa.

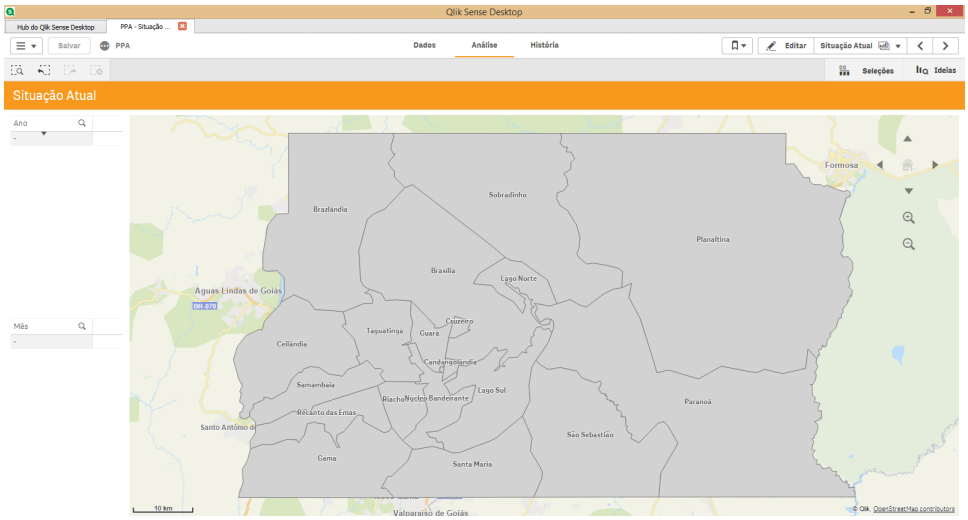


Figura 4.1 – Mapa do Distrito Federal

A figura 4.1 mostra as regiões do Distrito Federal que serão analisados no trabalho.

4.1 Exemplo: CEP 700 - Brasília

O CEP 700 contempla a região de Brasília e será utilizado para exemplificar o processamento dos dados. Além do CEP 700, essa região é composta dos CEPs 702,703,719,707 e 715. N resultado geral todos esses CEPs são considerados para analisar o preço dos imóveis naquela região.

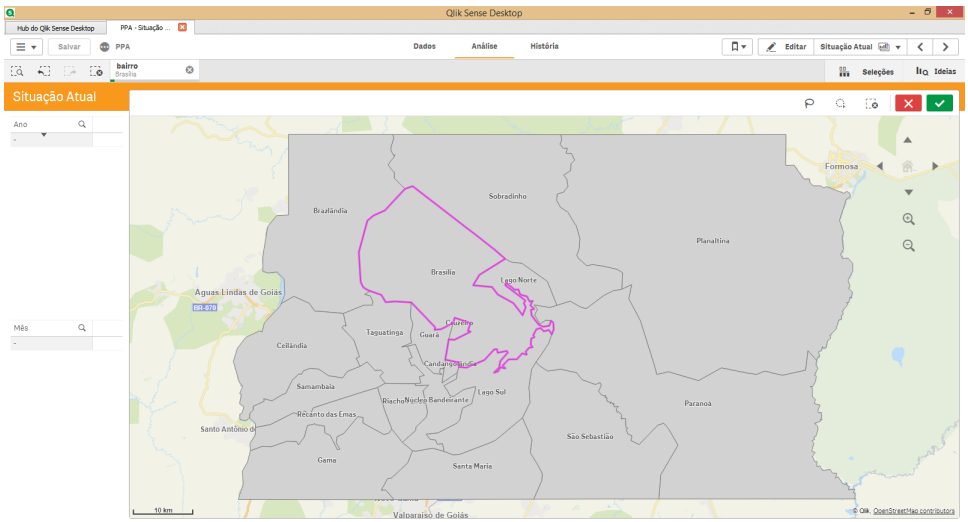


Figura 4.2 – Mapa de Brasília - Destaque de Brasília (CEP 700)

Os dados originais obtidos da Caixa Econômica Federal para o CEP 700 são mostrados de forma resumida na tabela 4.1. Nessa tabela são apresentados os primeiros três registros e os últimos três registros.

data_assinatura	cep_3	quantidade	media
23/01/1997	700	1	45.000,00
30/01/1997	700	1	90.000,00
13/02/1997	700	1	298.596,84
...	...	...	...
27/12/2011	700	2	350.000,00
28/12/2011	700	5	216.000,00
01/12/2011	700	10	332.250,00

Tabela 4.1 – Dados Originais (CEP 700)

A tabela 4.1 mostra os dados originais utilizados no trabalho.

A tabela utilizada para deflacionar a série de dados de imóveis será a da tabela 4.2. A base utilizada será 12/2011 para dar uma melhor perspectiva da evolução dos preços com valores mais atuais.

	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ	Acumulado
1997	1,58	0,42	1,16	0,59	0,3	0,7	0,09	-0,04	0,59	0,34	0,83	0,69	7,48%
1998	0,88	0,02	0,23	-0,13	0,23	0,28	-0,38	-0,17	-0,02	-0,03	-0,18	0,98	1,71%
1999	1,15	4,44	1,98	0,03	-0,34	1,02	1,59	1,45	1,47	1,89	2,53	1,23	19,99%
2000	1,02	0,19	0,18	0,13	0,67	0,93	2,26	1,82	0,69	0,37	0,39	0,76	9,80%
2001	0,49	0,34	0,8	1,13	0,44	1,46	1,62	0,9	0,38	1,45	0,76	0,18	10,40%
2002	0,19	0,18	0,11	0,7	1,11	1,74	2,05	2,36	2,64	4,21	5,84	2,7	26,41%
2003	2,17	1,59	1,66	0,41	-0,67	-0,7	-0,2	0,62	1,05	0,44	0,48	0,6	7,67%
2004	0,8	1,08	0,93	1,15	1,46	1,29	1,14	1,31	0,48	0,53	0,82	0,52	12,13%
2005	0,33	0,4	0,99	0,51	-0,25	-0,45	-0,4	-0,79	-0,13	0,63	0,33	0,07	1,22%
2006	0,72	-0,06	-0,45	0,02	0,38	0,67	0,17	0,41	0,24	0,81	0,57	0,26	3,79%
2007	0,43	0,23	0,22	0,14	0,16	0,26	0,37	1,39	1,17	0,75	1,05	1,47	7,89%
2008	0,99	0,38	0,7	1,12	1,88	1,89	1,12	-0,38	0,36	1,09	0,07	-0,44	9,10%
2009	0,01	-0,13	-0,84	0,04	0,18	-0,32	-0,64	0,09	0,25	-0,04	0,07	-0,11	-1,43%
2010	1,01	1,09	0,63	0,72	1,57	0,34	0,22	1,1	1,1	1,03	1,58	0,38	11,30%
2011	0,98	0,96	0,61	0,5	0,01	-0,13	-0,05	0,61	0,75	0,4	0,43	-0,16	5,01%

Tabela 4.2 – IGPDI - 1997 - 2011

Em seguida é realizado o tratamento dos dados com a utilização do deflator de IGPDI e o preenchimento dos dados faltantes com o valor anterior. Além disso, os *outliers* são substituídos pelo valor anterior. E por último, os dados são agrupados por CEP e a média do valor deflacionado é obtida, conforme tabela resumida 4.3.

cep	data	ano	mês	anoMes	mesAno	cep_anoMes	valor	valor sem outliers	inflacao	deflator	valorDeflacionado	outliers
700	35431	1997	1	1997/01	01/1997	700-1997/01	67.500,00	67.500,00	1,58	28,22	239.158,44	N
700	35462	1997	2	1997/02	02/1997	700-1997/02	183.799,76	183.799,76	0,42	28,68	640.929,47	N
700	35490	1997	3	1997/03	03/1997	700-1997/03	97.876,73	97.876,73	1,16	28,80	339.873,17	N
...	...	...	...	...	...	...	...	...	...	...	...	...
700	40817	2011	10	2011/10	10/2011	700-2011/10	260.881,47	260.881,47	0,40	99,17	263.060,34	N
700	40848	2011	11	2011/11	11/2011	700-2011/11	359.573,09	359.573,09	0,43	99,57	361.125,93	N
700	40878	2011	12	2011/12	12/2011	700-2011/12	307.583,11	307.583,11	-	100,00	307.583,11	N

Tabela 4.3 – Dados Tratados Agrupados (CEP 700)

Por fim, os dados são limpos e formatados para que possam ser utilizados no processamento da rede neural. Assim obtemos a tabela resumida 4.4.

data	valor
01/01/1997	239.158,44
01/02/1997	640.929,47
01/03/1997	339.873,17
...	...
01/10/2011	263.060,34
01/11/2011	361.125,93
01/12/2011	307.583,11

Tabela 4.4 – Dados Formatados (CEP 700)

Esses dados podem ser representados pela Figura 4.3.

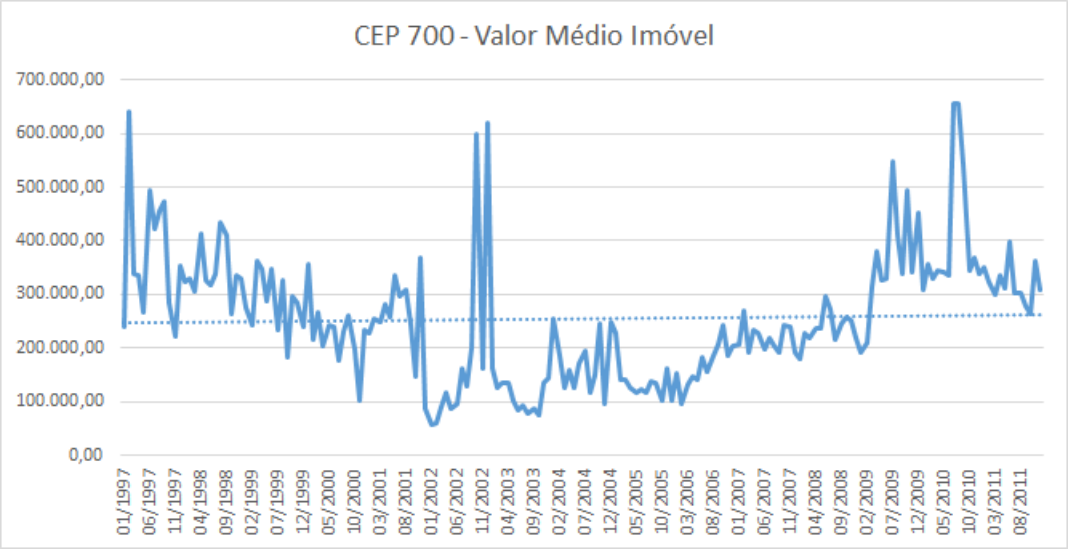


Figura 4.3 – Preço dos Imóveis do CEP 700 - Brasília entre 1997 e 2011

Os dados estão prontos para serem utilizados nas redes neurais *LSTM*. Antes de iniciar o processo de previsão, será desenvolvida uma linha de base de desempenho para o problema de previsão do projeto.

4.2 *Baseline – Random Walk*

O modelo de base ou *baseline* que será utilizado para a série temporal do trabalho em tela é uma previsão do tipo *random walk*. Essa previsão utiliza a observação do preço da etapa de tempo anterior (P_{t-1}) para prever o preço a observação na etapa de tempo atual (P_t).

$$P_t = P_{t-1} + \varepsilon \tag{4.1}$$

Onde ε é um erro aleatório com valor esperado zero, por isso o valor predito de P_t é igual a P_{t-1}.

Na figura 4.4 é mostrada a carga de dados do CEP 700 utilizando o modelo *random walk* na ferramenta *SPYDER*. Esse método utiliza os mesmos períodos de treinamento e testes que os demais métodos (Treinamento - 120, Testes - 60, Novas Previsões - 60). Os preços médios são mostrados no gráfico pela linha azul (180), já os valores de previsão de testes (60) são mostrados pela linha laranja e as novas previsões são mostradas pela linha vermelha (12). No caso do modelo de *baseline* as previsões utilizam o último valor da amostra e o repete para os demais períodos.

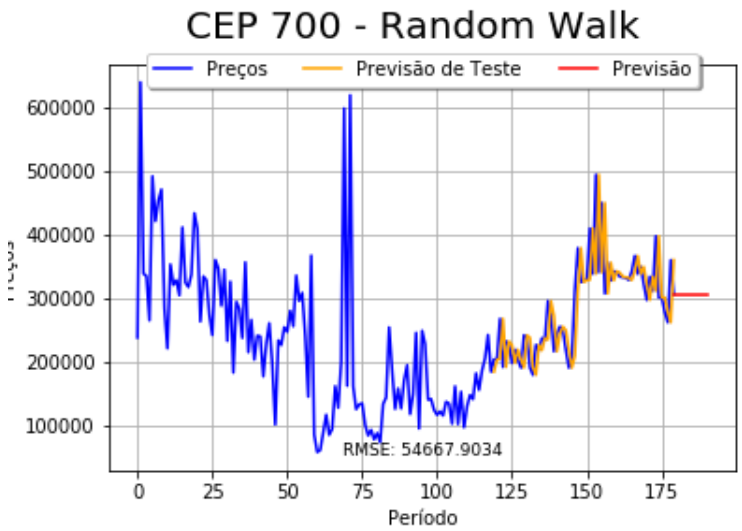


Figura 4.4 – Modelo de Baseline

Também pode ser representado pela tabela resumida 4.5.

	t-1	t+1
0	NA	239.158,44
1	239.158,44	640.929,47
2	640.929,47	339.873,17
3	339.873,17	335.930.64
4	335.930.64	266.130,62
...	...	...

Tabela 4.5 – Tabela da Baseline

A baseline apresenta os seguintes erros:

Score de Teste (*RMSE*):	54.667,90

A partir do momento que o *baseline* do conjunto de dados está completo, é possível começar a desenvolver o modelo *LSTM* para os dados do trabalho e comparar seus resultados.

4.3 Modelo LSTM

A rede *LSTM* é um tipo de rede neural recorrente que possui a vantagem de lembrar sequências longas. Com já mencionado anteriormente, o modelo utilizará a opção "*stateful*"do *keras*.

A camada *LSTM* espera que a entrada esteja em uma matriz com as dimensões: amostras (samples), etapas de tempo (*timesteps*) e características (*features*). A forma dos dados de entrada deve ser especificada na camada *LSTM* usando o argumento “*batch_input_shape*” como uma tupla que especifica o número esperado de amostras para ler cada lote, o número de etapas de tempo e o número de características.

O tamanho do lote é geralmente muito menor que o número total de amostras. Ele, juntamente com o número de épocas, define a rapidez com que a rede aprende os dados (com que frequência os pesos são atualizados). O tamanho do lote será de 1 e o número de épocas de 500.

O parâmetro de importação final na definição da camada *LSTM* é o número de neurônios, também chamado de número de unidades de memória ou blocos. Este é um problema razoavelmente simples e um número entre 1 e 5 deve ser suficiente. No trabalho em tela o número de neurônios utilizado foi de 1.

4.3.1 Modelo um passo a frente (*One Step Method*)

O primeiro passo é ajustar um modelo *LSTM* ao conjunto de dados existente, para isso, é necessário transformar os dados da série temporal em um problema de aprendizado supervisionado. Além disso, é importante transformar a série temporal em estacionária. Por último, as observações são transformadas para ter uma escala específica. Dessa forma, os dados são carregados na rede *LSTM*.

Na figura 4.5 é mostrada a carga de dados no ambiente de desenvolvimento *SPYDER*. Os preços médios dos imóveis com os ajustes já relatados são mostrados com a cor azul.

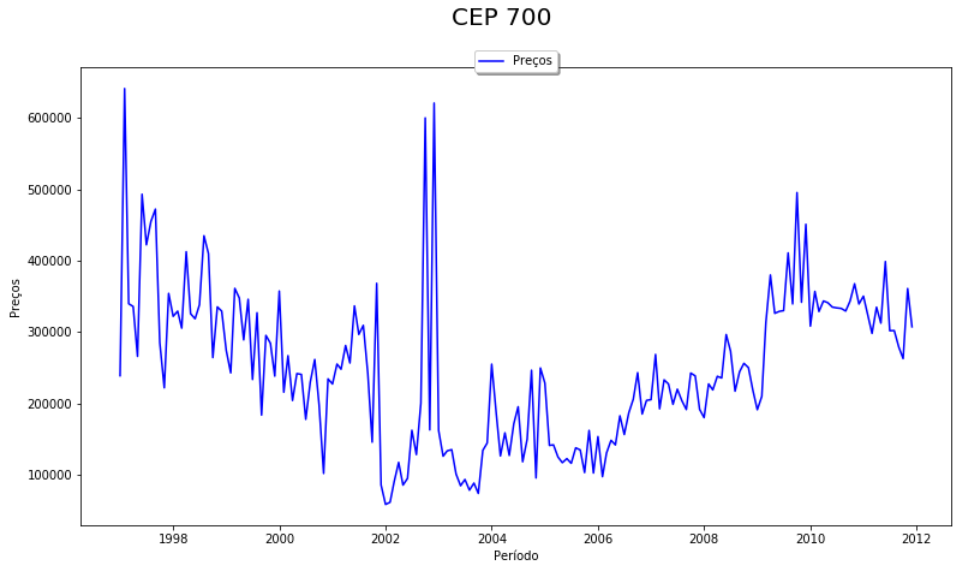


Figura 4.5 – Dados Carregados para Processamento

A ferramenta utilizada é conforme mostrada na figura 4.6.

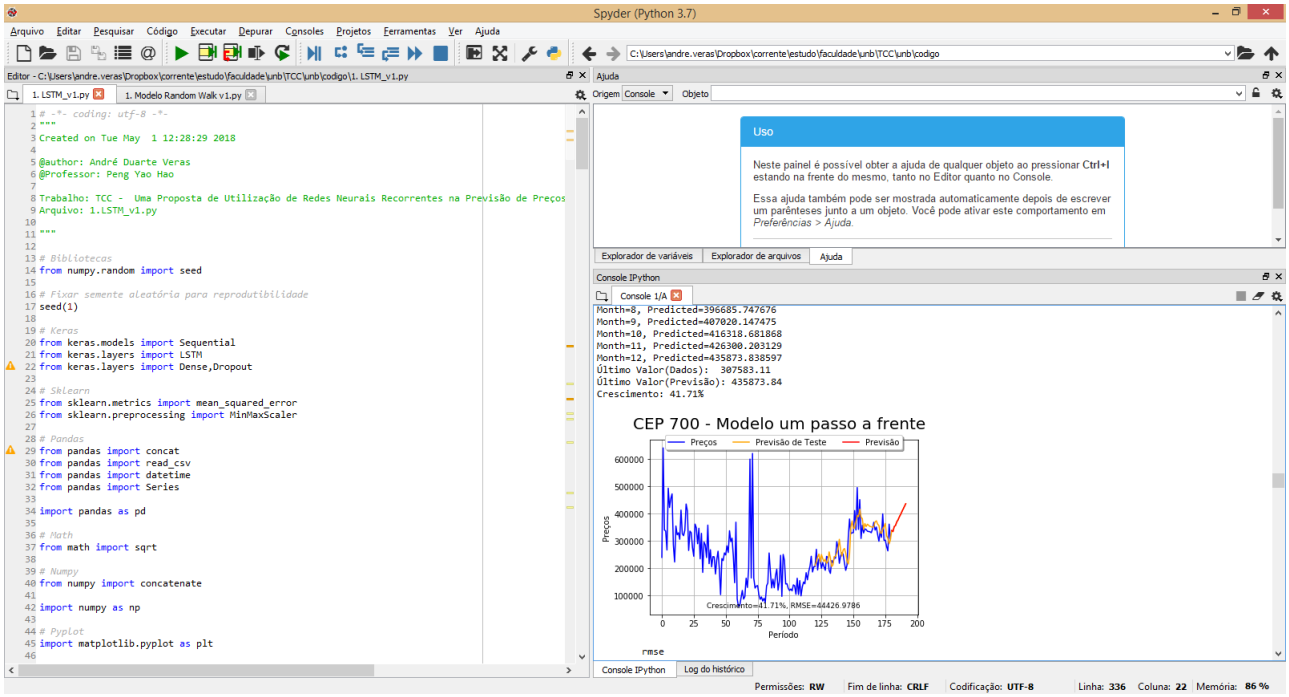


Figura 4.6 – Ferramenta SPYDER

O resultado utilizando o modelo de um passo a frente é representado pela figura 4.7.

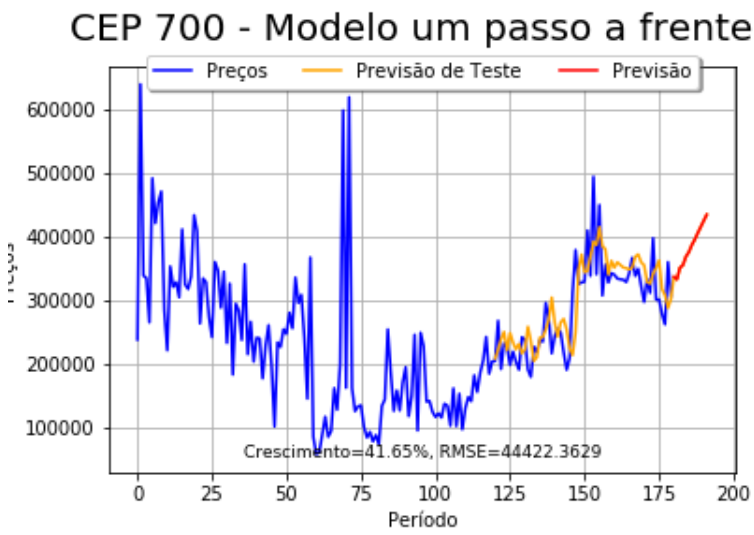


Figura 4.7 – Resultado um passo a frente

Também pode ser representado pela seguinte tabela resumida 4.6 que mostra as previsões no próximo ano (12 meses).

data	valor
01/01/2012	337.962,96
01/02/2012	333.686,97
01/03/2012	352.190,51
...	...
01/10/2012	416.186,28
01/11/2012	426.141,75
01/12/2012	435.695,86

Tabela 4.6 – Tabela de Resultados - Um passo a frente (CEP 700)

Esse modelo apresenta os seguintes erros e crescimento esperado para o próximo ano:

Score de Teste (RMSE):	44.422,36
Crescimento (1 ano):	41,65%

É possível notar uma melhora do valor em comparação ao valor do *baseline* de 54.667,90 *RMSE*. O modelo apresenta o crescimento dos preços médios de 41,65% do último período da amostra (Dezembro de 2011) até o último período de previsão (Dezembro de 2012).

4.3.2 Modelo de janela móvel (*Window Method*)

O resultado utilizando janela móvel é representado pela figura 4.8.

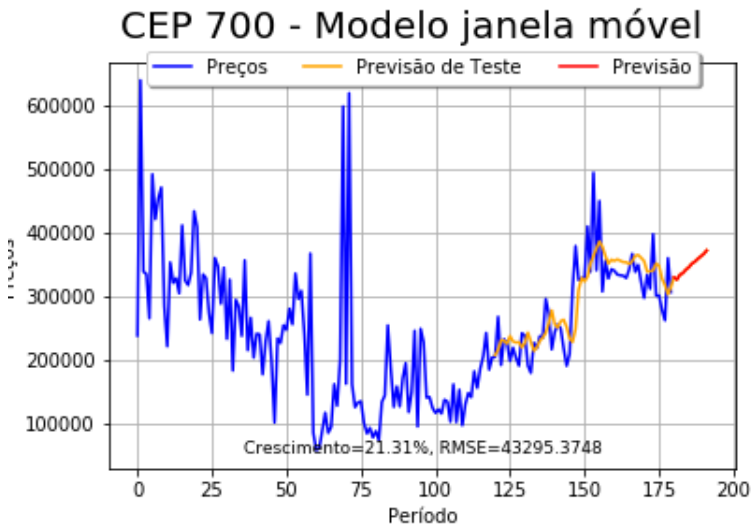


Figura 4.8 – Resultado Janela móvel

As previsões do próximo ano podem ser representadas pela seguinte tabela resumida 4.7.

data	valor
01/01/2012	330.811,66
01/02/2012	327.106,89
01/03/2012	334.888,38
...	...
01/10/2012	363.592,50
01/11/2012	367.641,58
01/12/2012	373.138,39

Tabela 4.7 – Tabela de Resultados - Janela móvel (CEP 700)

Esse modelo apresenta os seguintes erros:

Score de Teste (*RMSE*):	43.295,37
Crescimento (1 ano):	21,31%

Com o resultado é possível notar novamente uma melhora em relação aos resultados do modelo de um passo a frente. Sendo esse modelo o que apresentou o melhor resultado para

o CEP em análise. O próximo passo é executar o modelo para todos os CEPs e realizar uma média para buscar o método que apresenta o melhor score de teste. Já o crescimento dos preços apurado nesse modelo foi de 21,31% entre o último período da amostra até o último período de previsão.

O processo mostrado nessa seção representa apenas o conjunto de dados do CEP 700. Para representar todos os dados, foram executados o processos para cada CEP e plotados no mapa de Brasília.

4.4 Apresentação de Resultados

4.4.1 Resultados para CEP 700

A situação antes da previsão é mostrada pela figura 4.9.

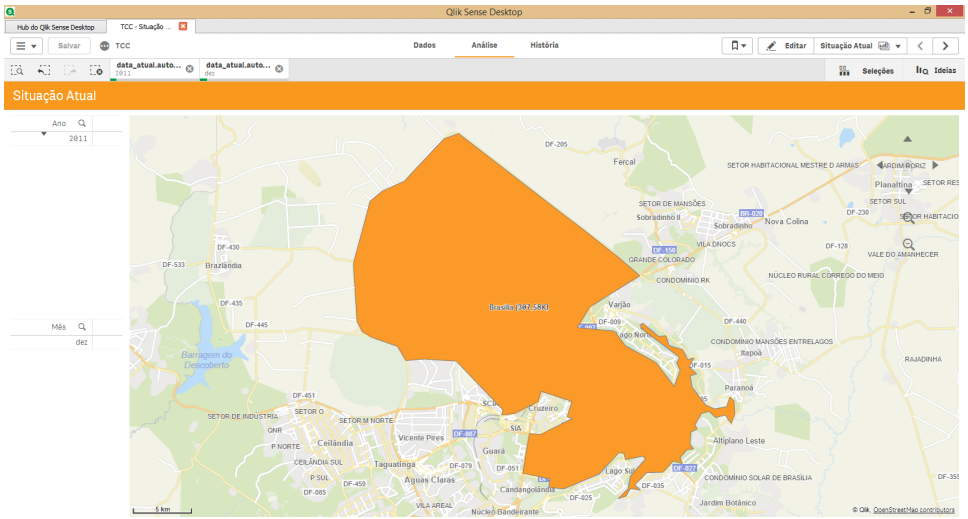


Figura 4.9 – Situação antes da previsão (Dezembro de 2011)

Na 4.9 é mostrado o preço médio do imóvel, do último mês da amostra de Dezembro de 2011 para o CEP 700.

A partir dessa situação, após a realização da previsão para um ano, os seguintes resultados são mostrados pela figura 4.10. Nela é possível ver a previsão de crescimento do preço do imóvel no mês de Dezembro de 2012, no valor de 41,65% no modelo de um passo a frente.

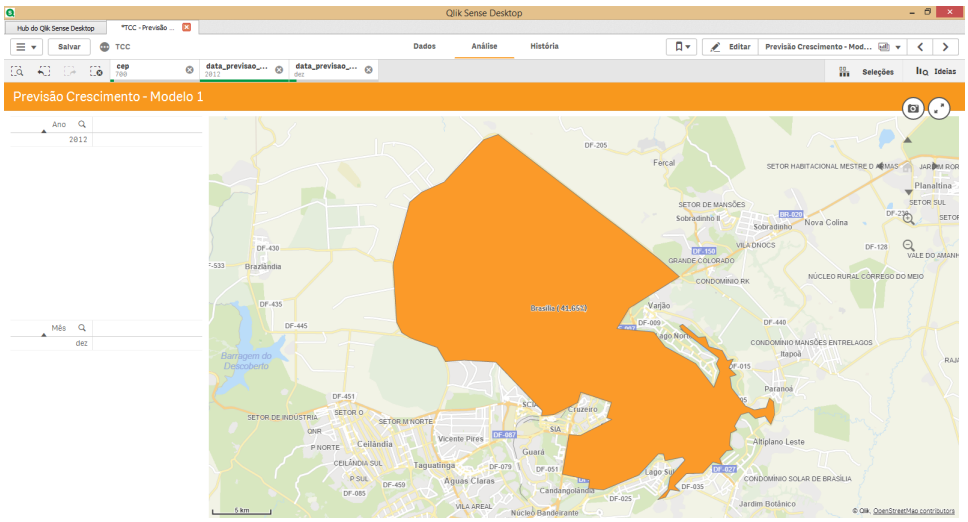


Figura 4.10 – Situação depois da previsão (Modelo de um passo a frente)

Para o modelo de previsão de janela foi obtido o resultado mostrado na 4.11 para o CEP 700, com o valor de previsão de crescimento em Dezembro de 2012, de 21,31%.

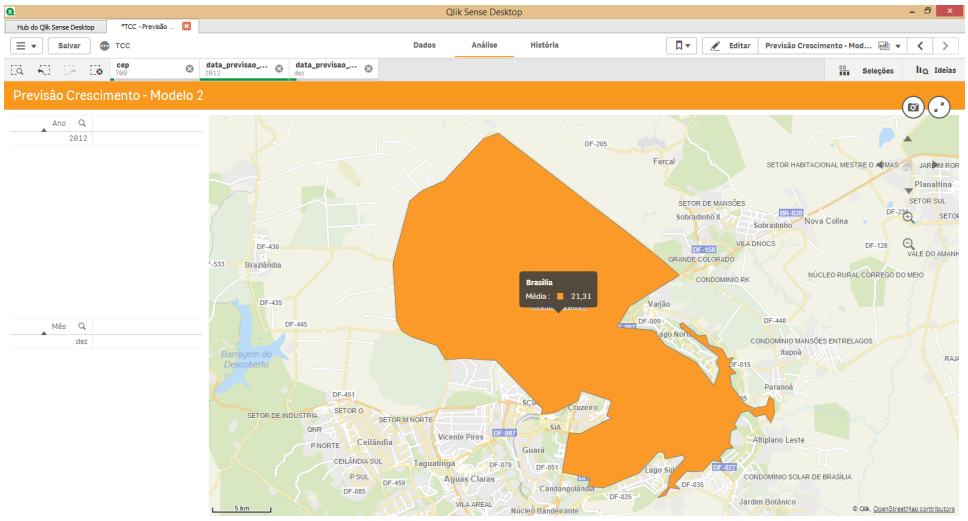


Figura 4.11 – Situação depois da previsão (Modelo de janela móvel)

Assim, tem-se os seguintes resultados em comparação com a *baseline* de acordo com a seguinte tabela para o CEP 700.

	RMSE	**Crescimento**
Modelo de Baseline	54.667,90	-
Modelo de um passo a frente	44.422,36	41,65%
Modelo de janela móvel	43.295,37	21,31%

4.4.2 Resultado Geral

Nessa seção será mostrado o resultado geral de todos os bairros de Brasília. A situação antes da previsão pode ser mostrada para o último período da amostra de Dezembro de 2011, conforme figura 4.12. No gráfico são mostrados a média de preços por região do Distrito Federal, considerando a média dos preços dos CEPs correspondentes a essa região. As regiões mais escuras são as que apresentam as maiores médias.

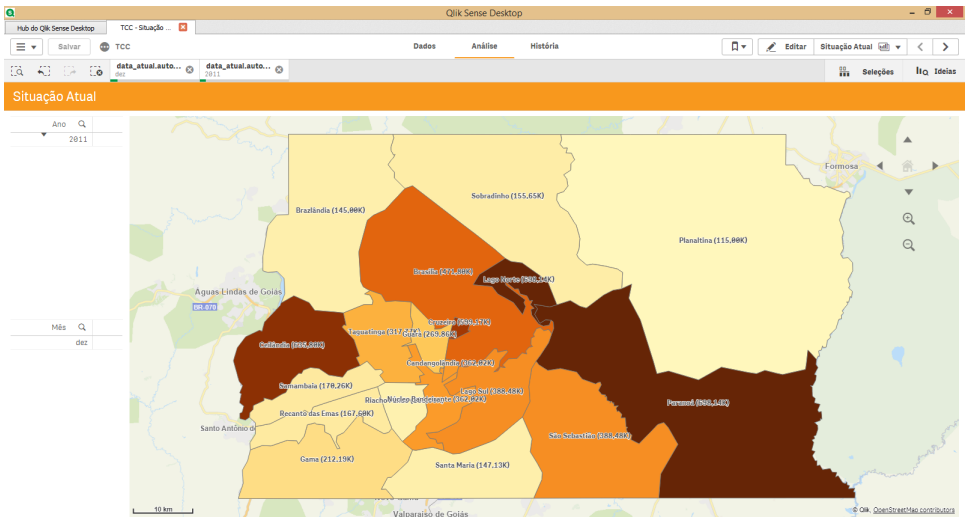


Figura 4.12 – Situação do preço médio do último período da amostra antes da previsão

Primeiramente, utilizou-se o método um passo a frente e foram obtidos os seguintes resultados.

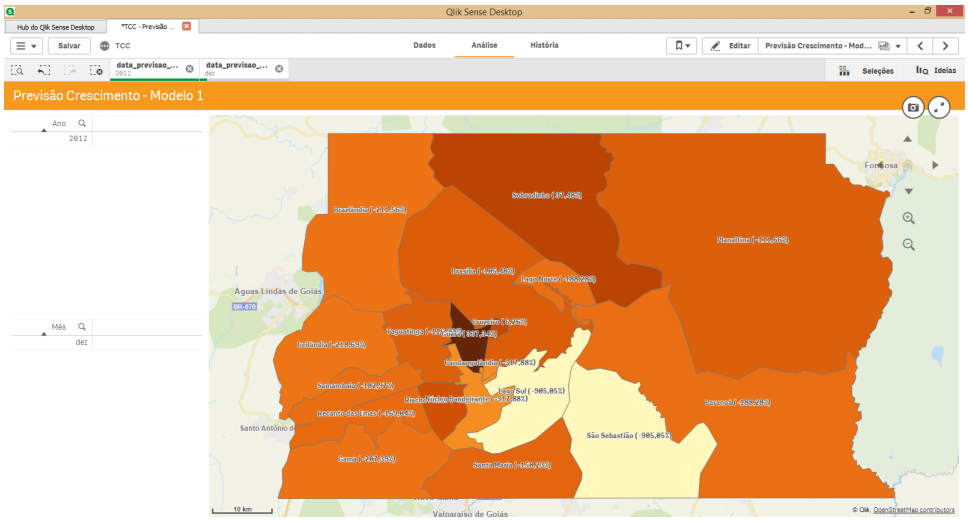


Figura 4.13 – Previsão de crescimento dos preços de imóveis para o Distrito Federal - Modelo de um passo a frente

Na figura 4.13 é possível ver quais regiões que tendem a ter um crescimento maior de preço

em Dezembro de 2012 em comparação com outras regiões com base em Dezembro de 2011. Ou seja, foi realizado o cálculo de crescimento um ano a partir do último registro obtido.

O próximo passo foi utilizar o segundo método de janela móvel, e os resultados foram os seguintes.

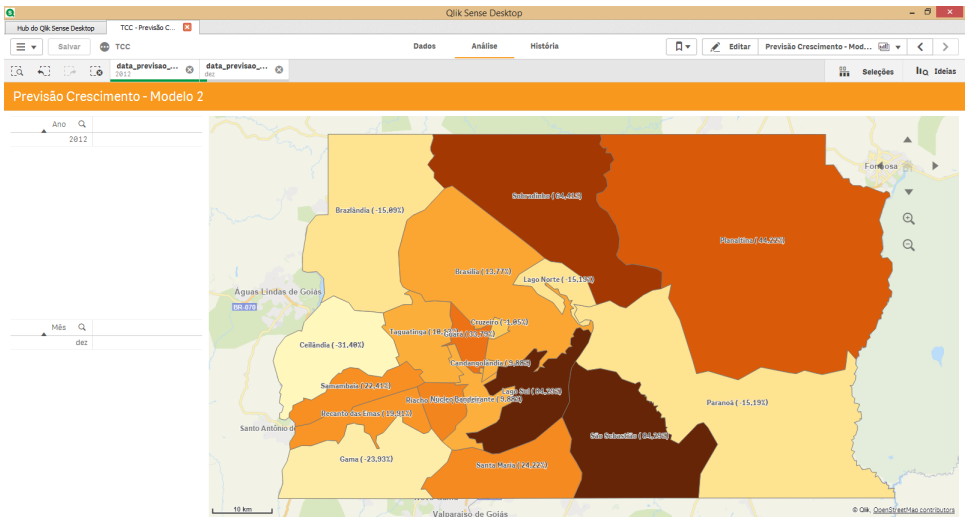


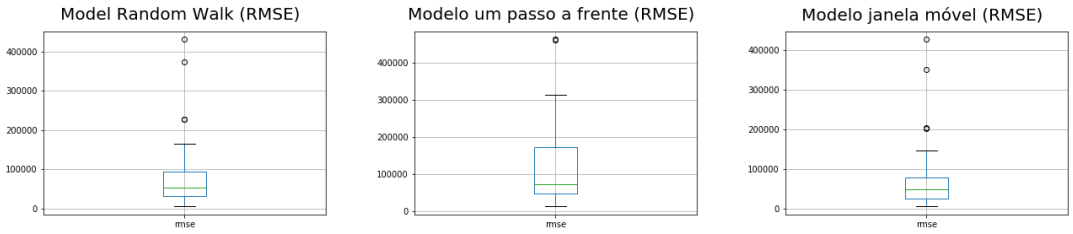
Figura 4.14 – Previsão de crescimento dos preços de imóveis para o Distrito Federal - Modelo de janela móvel

Na figura 4.14 é possível ver quais as regiões que tendem a ter um crescimento maior de preço no mês de Dezembro de 2012, em comparação com outras regiões.

O resultado da métrica *RMSE* utilizada para avaliar os modelos é dado pela tabela abaixo.

	***Random Walk* (a)**	**Um passo a frente (b)**	**Janela móvel (c)**
Número	26	26	26
Média	95.168,65	121.590.45	85.573,53
Desvio Padrão	107.886,83	124.836,71	104.112,72
Mínimo	5.872,49	12.396,90	6.303,90
25%	32.119,93	47.427,79	25.478,24
50%	53.344,53	72.979,83	48.565,13
75%	94.831,30	172.289,10	77.887,48
Máximo	430.655,35	463.190,28	426.821,12

Esse resultado pode ser mostrado graficamente como a figura 4.15.



(a) (b) (c)

Figura 4.15 – Gráfico dos Resultados Gerais

O resultado mostra que o modelo de janela móvel, no geral, também apresentou um resultado melhor que o modelo de um passo a frente e o modelo de *baseline*.

5 CONCLUSÃO

Esta pesquisa teve por objetivo colocar em prática o uso de redes neurais recorrentes em um processo de previsão de preços de imóveis, de uma base de dados real da região do Distrito Federal. No trabalho foi apresentado um método para previsão de preços de imóveis que pode ser facilmente replicado para outras regiões do Brasil.

Na pesquisa foram utilizados dados apenas da região do Distrito Federal, porém o estudo poderia englobar o nível de estado ou até mesmo de país. Além disso, poderia ser replicado para outras regiões. No trabalho foi utilizado apenas o preço como variável explicativa para a previsão. Ou seja, utilizou-se um modelo univariado. Porém, existem outras variáveis como o próprio tempo (número de dias), e variáveis macroeconômicas como inflação, estoque do crédito, taxa de desemprego e produto interno bruto (PIB) que podem influir no preço dos imóveis.

A partir da organização dos dados coletados, tratados de forma que pudessem ser utilizados no trabalho, as análises realizadas mostraram que o modelo que apresentou o melhor resultado foi o modelo de janela móvel (12 períodos de tempo) em relação ao de um passo a frente (1 período de tempo) e o modelo de baseline Random Walk. As regiões do Distrito Federal que apresentaram os maiores crescimentos, segundo o modelo de janela móvel em dezembro de 2012 (um ano após os últimos registros da amostra) foram na ordem: Lago Sul e São Sebastião (84,29%), Sobradinho (64,41%) e Planaltina (44,22%).

Uma possibilidade futura de trabalho na área seria utilizar dados mais recentes e também uma forma de comprovar se as previsões apresentadas nesse trabalho pelos modelos se confirmaram para os próximos anos. Também poderia ser realizada uma coleta de dados em larga escala e obter dados de outros estados ou até mesmo do país como um todo para aplicação do método. Além disso, poderiam ser utilizadas outras configurações dos hiperparâmetros como o número de neurônios, número de passos, número de épocas, número de camadas e estrutura da LSTM, modelo stateless e tamanho do lote. Variando esses hiperparâmetros seria possível identificar qual apresentaria melhores resultados.

REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO, E. G.; PEREIRA, J. C.; XIMENES, F.; SPANHOL, C. P.; GARSON, S. Proposta de uma metodologia para a avaliação do preço de venda de imóveis residenciais em bonito/ms baseado em modelos de regressão linear múltipla. *P&D em Engenharia de Produção*, v. 10, n. 2, p. 195–207, 2012.

BAHIA, I. S. H. A data mining model by using ann for predicting real estate market: Comparative study. *International Journal of Intelligence Science*, Scientific Research Publishing, v. 3, n. 04, p. 162, 2013.

BALARINE, O. F. O. Contribuições macroeconômicas ao entendimento da formação de preços habitacionais locais. *ENCONTRO NACIONAL DE ENGENHARIA DA PRODUÇÃO*, ENEGEP Piracicaba, v. 15, 1997.

BAPTISTELLA, M. O uso de redes neurais e regressão linear múltipla na engenharia de avaliações: Determinação dos valores venais de imóveis urbanos. *Diss., Universidade Federal do Paraná*, 2005.

BRONDINO, N. C. M. Estudo da influência da acessibilidade no valor de lotes urbanos através do uso de redes neurais. *São Carlos–SP*, 1999.

CAPLIN, A.; CHOPRA, S.; LEAHY, J.; LECUN, Y.; THAMPY, T. Machine learning and the spatial structure of house prices and housing returns. 2008.

CHATFIELD, C. *The analysis of time series: an introduction*. [S.l.]: Chapman and Hall/CRC, 2016.

DANTAS, R. A.; CORDEIRO, G. Uma nova metodologia para avaliação de imóveis utilizando modelos lineares generalizados. *Revista Brasileira de Estatística*, v. 49, n. 191, p. 27–46, 1988.

DOLHANSKY, B. *Artificial Neural Networks: Linear Regression (Part 1)*. 2013. Disponível em: <http://www.briandolhansky.com/blog/artificial-neural-networks-linear-regression-part-1>.

FACURE, M. *Funções de Ativação*. 2017. Disponível em: <https://matheusfacure.github.io/2017/07/12/activ-func/>.

GONZÁLEZ, M. A. S. Análise da evolução recente dos aluguéis em porto alegre. *Anais do XVII Encontro Nacional de Engenharia de Produção–ENEGEP*, 1997.

GONZÁLEZ, M. A. S. Aplicação de técnicas de descobrimento de conhecimento em bases de dados e de inteligência artificial em avaliação de imóveis. 2002.

GONZÁLEZ, M. A. S.; FORMOSO, C. T. Análise conceitual das dificuldades na determinação de modelos de formação de preços através de análise de regressão. n. 8, p. 65–75, 2000.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. http://www.deeplearningbook.org.

GUEDES, J. C. Aplicação de redes neurais na avaliação de bens. **I Seminário Internacional da LARES (I LARES International meeting) São Paulo**. [S.l.: s.n.], 1999.

KHAMIS, A. B.; KAMARUDIN, N. K. K. B. Comparative study on estimate house price using statistical and neural network model. *International Journal of Scientific & Technology Research*, v. 3, n. 12, p. 126–131, 2014.

LEE, T.-W.; CHEN, K. Prediction of house unit price in taipei city using support vector regression. *APIEMS2016, Taipei City, Republic of China*, v. 12, n. 7, 2016.

LIM, W. T.; WANG, L.; WANG, Y.; CHANG, Q. Housing price prediction using neural networks. **Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12th International Conference on**. [S.l.], 2016. p. 518–522.

LIMSOMBUNCHAI, V. House price prediction: hedonic price model vs. artificial neural network. **New Zealand Agricultural and Resource Economics Society Conference**. [S.l.: s.n.], 2004. p. 25–26.

LIN, H.; CHEN, K. Predicting price of taiwan real estates by neural networks and support vector regression. **Proc of the 15th WSEAS international Conference**. [S.l.: s.n.], 2010.

LIN, H.-Y.; CHEN, K. The trend of average unit price in taipei city. *Research in World Economy*, v. 6, n. 1, p. 133, 2015.

LIPTON, Z. C. *A Critical Review of Recurrent Neural Networks for Sequence Learning*. 2015. Disponível em: <https://www.groundai.com/project/a-critical-review-of-recurrent-neural-networks-for-sequence-learning/>.

MACHADO, R. L.; HEINECK, L. F. Modelagem econométrica de preços de apartamentos residenciais de três dormitórios em florianópolis–sc. *Anais do XVIII Encontro Nacional de Engenharia de Produção–ENEGEP*, 1998.

MOREIRA, D. S.; SILVA, R. dos S.; FERNANDES, A. M. da R. Avaliação de imóveis utilizando análise multicritério e redes neurais artificiais. 2010.

MU, J.; WU, F.; ZHANG, A. Housing value forecasting based on machine learning methods. **Abstract and Applied Analysis**. [S.l.], 2014. v. 2014.

NETO, A. P. Avaliação de imóveis urbanos com utilização de sistemas nebulosos (redes neuro-fuzzy) e redes neurais artificiais. **Congresso Panamericano de Valuación, XXI**. [S.l.: s.n.], 2004.

NG, A.; DEISENROTH, M. *Machine learning for a london housing price prediction mobile application*. [S.l.], 2015.

NGHIEP, N.; AL, C. Predicting housing value: A comparison of multiple regression analysis and artificial neural networks. *Journal of real estate research*, American Real Estate Society, v. 22, n. 3, p. 313–336, 2001.

NIELSEN, M. *Why are deep neural networks hard to train?* 2018. Disponível em: <http://neuralnetworksanddeeplearning.com/chap5.html>.

NIELSEN, M. *Why are deep neural networks hard to train?* 2018. Disponível em: <http://neuralnetworksanddeeplearning.com/chap5.html>.

PARK, B.; BAE, J. K. Using machine learning algorithms for housing price prediction: The case of fairfax county, virginia housing data. *Expert Systems with Applications*, Elsevier, v. 42, n. 6, p. 2928–2934, 2015.

PLAKANDARAS, V.; GUPTA, R.; GOGAS, P.; PAPADIMITRIOU, T. Forecasting the us real house price index. *Economic Modelling*, Elsevier, v. 45, p. 259–267, 2015.

POW, N.; JANULEWICZ, E.; LIU, L. *Applied Machine Learning Project 4 Prediction of real estate property prices in Montréal*. 2014.

SANTOS, L. A. dos. *Artificial Intelligence*. 2018. Disponível em: <https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/machine$_{l}earning.html$>.

SAUNDERS, C.; GAMMERMAN, A.; VOVK, V. Ridge regression learning algorithm in dual variables. 1998.

SELIM, S. Determinants of house prices in turkey: A hedonic regression model. *Doğuş Üniversitesi Dergisi*, v. 9, n. 1, p. 65–76, 2011.

SILVA, J. A. M.; NALI, L. R.; MAROTE, F. M. M. Modelagem de dados por regressão linear múltipla para avaliação de imóveis rurais do submédio são francisco. **XV Congresso Brasileiro de Engenharia de Avaliações e Perícias**. [S.l.: s.n.], 2009.

TEIXEIRA, M. C. C.; OCERÍN, J. Caridad y. Variáveis explicativas e a sua importância na formação do preço de um apartamento em portugal: uma abordagem com redes neuronais artificiais. *XXI Jornadas Hispano-Lusas de Gestión Científica, Cordoba, 2-4 Fev. 2011*, 2011.

TRAWIŃSKI, B.; TELEC, Z.; KRASNOBORSKI, J.; PIWOWARCZYK, M.; TALAGA, M.; LASOTA, T.; SAWIŁOW, E. Comparison of expert algorithms with machine learning models for real estate appraisal. **INnovations in Intelligent SysTems and Applications (INISTA), 2017 IEEE International Conference on**. [S.l.], 2017. p. 51–54.

WU, J. Y. Housing price prediction using support vector regression. 2017.

YOO, S.; IM, J.; WAGNER, J. E. Variable selection for hedonic model using machine learning approaches: A case study in onondaga county, ny. *Landscape and Urban Planning*, Elsevier, v. 107, n. 3, p. 293–306, 2012.

6 PROGRAMAÇÕES UTILIZADAS

```
# -*- coding: utf-8 -*-
"""
Created on Tue May  1 12:28:29 2018

@author: Andr  Duarte Veras
@Professor: Peng Yao Hao

Trabalho: TCC -  Uma Proposta de Utiliza  o de Redes Neurais
   Recorrentes na Previs o de Pre os de Im veis no Distrito
   Federal
Arquivo: 0. Modelo Random Walk

"""
# Bibliotecas

# Internacionaliza  o
import locale

# numpy
import numpy

# time
import time

# math
from math import sqrt

# pandas
from pandas import options
from pandas import read_csv
from pandas import datetime
from pandas import DataFrame
from pandas import concat
from matplotlib import pyplot
from sklearn.metrics import mean_squared_error

# os
import os
```

```
# Fun   o para serparar as sequ ncias
def parser(x):
        return datetime.strptime(x, '%d/%m/%Y')

# Fun   o de Modelo de Persist ncia
def model_persistence(x):
        return x

# Fun   o para previs o por CEP
def principal(cep: str):

    print
       ('------------------------------------------------------------------

    print('In cio')
    print
       ('------------------------------------------------------------------



    print
       ('------------------------------------------------------------------

    print('Configura  es Iniciais')
    print
       ('------------------------------------------------------------------


    # Marca in cio do processamento para contagem do tempo
    start_time = time.time()

    # Sa da em formato portugu s do Brasil
    locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')

    # Ao imprimir vetor coloca separador de milhar e decimal
       portugu s do Brasil
    options.display.float_format = '{:n}'.format

    # Lembrar de remover espa os vazios e caracteres no fim do
        arquivo
    # Carregar arquivos de CEPs
    dados = read_csv(cep + 'F.csv', header=0, parse_dates=[0],
       date_parser=parser, delimiter='\t', decimal=',',
       thousands='.')

    print
       ('------------------------------------------------------------------
```

```
print('Vetor: dados')
print(dados)
print('Tamanho (dados):' + str(dados.shape))
print
   ('------------------------------------------------------------------


# Criar um conjunto de dados atrasado
dados_valor = DataFrame(dados, columns=['valor'])
dados_atraso = concat([dados_valor.shift(1), dados_valor],
   axis=1)
dados_atraso.columns = ['t-1', 't+1']

print
   ('------------------------------------------------------------------

print('Vetor: dados_atraso')
print(dados_atraso)
print('Tamanho (dados_atraso):' + str(dados_atraso.shape))
print
   ('------------------------------------------------------------------


# Remove o NaN que foi gerado pela diferen a na primeira
   linha
dados_supervisionado = dados_atraso.iloc[1:]

# Transforma Dataframe em array
dados_valor = dados_valor.values

# Busca apenas valores no array
dados_supervisionado = dados_supervisionado.values

# Atributos da amostra, treinamento e teste, previsao
n_amostra = 180 #12 #180
n_treinamento = 120 #8 #120
n_teste = 60 #4 #60
n_previsao = 12 #12 #36
posicao_rmse = 50000 #10 # 50000

# Dividir em conjuntos de treinamento e testes
train, test = dados_supervisionado[0:n_treinamento - 2],
   dados_supervisionado[n_treinamento-2:n_treinamento +
   n_teste - 1]

print(train)
print(test)
print('Treinamento: %s, Teste: %s' % (len(train), len(test)
```

```
   ))
print
   ('-----------------------------------------------------------------


train_X, train_y = train[:,0], train[:,1]
test_X, test_y = test[:,0], test[:,1]

print(train_X)
print(train_y)
print('Treinamento - train_X: %s, train_y: %s' % (train_X.
   shape, train_y.shape))
print
   ('-----------------------------------------------------------------


print(test_X)
print(test_y)
print('Teste - test_X: %s, test_y: %s' % (test_X.shape,
   test_y.shape))
print
   ('-----------------------------------------------------------------


# Pervis es de Teste
dados_previsao_teste = list()
for x in test_X:
    yhat = model_persistence(x)
    dados_previsao_teste.append(yhat)

print
   ('-----------------------------------------------------------------

print('Vetor: dados_previsao_teste')
print(numpy.array(dados_previsao_teste))
print('Tamanho (dados_previsao_teste):' + str(numpy.array(
   dados_previsao_teste).shape))
print
   ('-----------------------------------------------------------------


print
   ('-----------------------------------------------------------------

print('Imprime Performance RMSE')
print
   ('-----------------------------------------------------------------
```

```
    print(dados_valor[n_treinamento:n_treinamento + n_teste])
    print(dados_previsao_teste[1:])

    rmse = sqrt(mean_squared_error(dados_valor[n_treinamento:
       n_treinamento + n_teste], dados_previsao_teste[1:]))
    print('RMSE: %.4f' % rmse)
    print
       ('-----------------------------------------------------------------


    # Salvar RMSE em arquivo
    saida_rmse = DataFrame([[cep,rmse]], columns = ['cep', '
       rmse'])
    saida_rmse.to_csv(r'rmse_random_walk.csv', header=None,
       index=None, sep=';', mode='a', decimal=',')

    # Previs es

    # Busca o  ltimo  valor da amostra e repete ele nas
       previs es.
    ultima_valor = dados_valor[-1]
    dados_previsao = list()
    for x in range(n_treinamento + n_teste , n_treinamento +
       n_teste + n_previsao):
        dados_previsao.append(ultima_valor)

    print
       ('-----------------------------------------------------------------

    print('Vetor: dados_previsao')
    print(numpy.array(dados_previsao))
    print('Tamanho (dados_previsao):' + str(numpy.array(
       dados_previsao).shape))
    print
       ('-----------------------------------------------------------------


    print
       ('-----------------------------------------------------------------

    print('Imprime Gr fico')
    print
       ('-----------------------------------------------------------------


    # Para imprimir gr fico grande
#    pyplot.figure(figsize=(13,7))
```

```
#    pyplot.xticks(numpy.arange(0, n_amostra + n_previsao, step
   =12.0))

    pyplot.suptitle('CEP '  + cep + ' - Random Walk', fontsize
       =20)
    pyplot.ylabel('Pre os')
    pyplot.xlabel('Per odo')
    pyplot.plot(dados.valor, color='blue', label='Pre os')
    pyplot.plot([None for _ in range(n_treinamento - 1)] +
                [x for x in dados_previsao_teste], color='
                   orange', label = 'Previs o de Teste')
    pyplot.plot([None for _ in range(n_treinamento + n_teste -
       1)] +
                [x for x in dados_previsao], color='red',
                   label='Previs o')
    pyplot.grid()
    ax = plt.gca()
    pyplot.text(0.5, 0.05, 'RMSE: %.4f' % rmse , fontsize=9, ha
       ='center', va='center', transform = ax.transAxes)
    pyplot.legend(loc='upper center', bbox_to_anchor=(0.5,
       1.05), ncol=3, fancybox=True, shadow=True)
    pyplot.savefig(cep + '_random_walk.png')
    pyplot.show()


    print
       ('-----------------------------------------------------------------

    print('Imprime tempo de processamento')
    print
       ('-----------------------------------------------------------------

    elapsed_time = time.time() - start_time
    print('Tempo do CEP '+cep + ': ' + time.strftime("%H:%M:%S
       ", time.gmtime(elapsed_time)))
    print
       ('-----------------------------------------------------------------



    return rmse

# Fun   o principal para carga de Ceps
def main():

    # Remove arquivo de rmse
    if os.path.exists("rmse_random_walk.csv"):
        os.remove("rmse_random_walk.csv")
```

```
    ceps = ['700','702','703','706','707','710',
            '711','712','715','716','717','718',
            '719','720','721','722','723','724',
            '725','726','727','730','731','732',
            '733','734']

    #ceps = ['700']

    error_scores = list()

    for cep in ceps:
        rmse = principal(cep)
        error_scores.append(rmse)

    # Resumo resultdos
    results = DataFrame()
    results['rmse'] = error_scores
    print(results.describe())
    results.describe().to_csv("resumo_modelo_random_walk.csv")
    results.boxplot()
    pyplot.suptitle('Model Random Walk (RMSE)', fontsize=20)
    pyplot.savefig('resumo_modelo_random_walk.png')
    pyplot.show()

if __name__ == '__main__':
    main()



   # -*- coding: utf-8 -*-
"""
Created on Tue May  1 12:28:29 2018

@author: Andr  Duarte Veras
@Professor: Peng Yao Hao

Trabalho: TCC -  Uma Proposta de Utiliza  o de Redes Neurais
   Recorrentes na Previs o de Pre os de Im veis no Distrito
   Federal
Arquivo: 1.LSTM_v1.py

"""

# Bibliotecas
from numpy.random import seed

# Fixar semente aleat ria para reprodutibilidade
```

```
seed(1)

# Keras
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense,Dropout

# Sklearn
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import MinMaxScaler

# Pandas
from pandas import concat
from pandas import read_csv
from pandas import datetime
from pandas import Series

import pandas as pd

# Math
from math import sqrt

# Numpy
from numpy import concatenate

import numpy as np

# Pyplot
import matplotlib.pyplot as plt

# Outros
import locale
import time
import os
import winsound

# fun  o de an lise de data para carregar o conjunto de
   dados
def parser(x):
        return datetime.strptime(x, '%d/%m/%Y')

# crie uma s rie de diferen as dos valores
def difference(dataset, interval=1):
        diff = list()
        for i in range(interval, len(dataset)):
                value = dataset[i] - dataset[i - interval]
                diff.append(value)
        return Series(diff)
```

```
# valor diferenciado invertido
def inverse_difference(history, yhat, interval=1):
    return yhat + history[-interval]

# escala de dados para [-1, 1]
def scale(data, scaler=None):
    if scaler is None:
        # fit scaler
        scaler = MinMaxScaler(feature_range=(-1, 1))
        scaler = scaler.fit(data)
    # transform
    data = data.reshape(data.shape[0], data.shape[1])
    data_scaled = scaler.transform(data)
    return scaler, data_scaled

# escala inversa para um valor previsto
def invert_scale(scaler, X, value):
        new_row = [x for x in X] + [value]
        array = np.array(new_row)
        array = array.reshape(1, len(array))
        inverted = scaler.inverse_transform(array)
        return inverted[0, -1]


## ajuste uma rede LSTM aos dados de treinamento
def fit_lstm(train, batch_size, nb_epoch, neurons, n_steps,
   n_features):
    X, y = train[:, 0:-1], train[:, -1]
    X = X.reshape(X.shape[0], X.shape[1],1 )

    #-----------------------------------------------------------------

    # Modelo 1
    model = Sequential()
    model.add(LSTM(neurons[0], batch_input_shape=(batch_size, X
       .shape[1], X.shape[2]), stateful=True))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    '''
    # Modelo 2
    model = Sequential()
    model.add(LSTM(neurons[0], batch_input_shape=(batch_size, X
       .shape[1], X.shape[2]), stateful=True,return_sequences=
       True))
    model.add(Dropout(0.3))
    model.add(LSTM(neurons[1], batch_input_shape=(batch_size, X
       .shape[1], X.shape[2]), stateful=True))
```

```
    model.add(Dropout(0.3))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    '''
    '''
    # Modelo 3 -  com relu (N o precisa de scale)
    model = Sequential()
    model.add(LSTM(50, activation='relu', input_shape=(n_steps,
        n_features)))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mse')
    '''
    '''
    # Modelo 4
    model = Sequential()
    model.add(LSTM(neurons[0], batch_input_shape=(batch_size,
       n_steps, n_features), stateful=True, return_sequences=
       True))
    model.add(LSTM(neurons[1], batch_input_shape=(batch_size,
       n_steps, n_features), stateful=True))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    '''
    #----------------------------------------------------------------

    for i in range(nb_epoch):
        model.fit(X, y, epochs=1, batch_size=batch_size,
           verbose=0, shuffle=False)
        model.reset_states()
    return model

# atualiza modelo LSTM
def update_model(model, train, batch_size, updates):
        X, y = train[:, 0:-1], train[:, -1]
        X = X.reshape(X.shape[0], X.shape[1],1 )
        for i in range(updates):
                model.fit(X, y, epochs=1, batch_size=batch_size
                   , verbose=0, shuffle=False)
                model.reset_states()

# converter uma matriz de valores em uma matriz de conjunto de
   dados
def create_dataset(dataset, look_back=1):
        dataset = np.insert(dataset,[0]*look_back,0)
        dataX, dataY = [], []
        for i in range(len(dataset)-look_back):
                a = dataset[i:(i+look_back)]
                dataX.append(a)
```

```
                dataY.append(dataset[i + look_back])
        dataY= np.array(dataY)
        dataY = np.reshape(dataY,(dataY.shape[0],1))
        dataset = np.concatenate((dataX,dataY),axis=1)
        return dataset

# fa a uma previs o de um passo
def forecast_lstm(model, batch_size, X, n_steps, n_features):
    X = X.reshape((1, n_steps, n_features))
    yhat = model.predict(X, batch_size=batch_size)
    return yhat[0, 0]

# computar RMSPE
def RMSPE(x,y):
        result=0
        for i in range(len(x)):
                result += ((x[i]-y[i])/x[i])**2
        result /= len(x)
        result = sqrt(result)
        result *= 100
        return result


def principal(cep, nome_modelo, numero_modelo,  n_steps):

    # Varia is do modelo
    #-----------------------------------------------------------------

    #n_steps = 1
    n_epoch = 200 #200 #1500
    neurons = [1,1]
    n_features = 1
    n_previsao = 12
    updates = 1
    #-----------------------------------------------------------------


    # Sa da em formato portugu s do Brasil
    locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')

    # Ao imprimir vetor coloca separador de milhar e decimal
       portugu s do Brasil
    pd.options.display.float_format = '{:n}'.format

    # Carregar dados
    series = read_csv(cep + 'F.csv', header=0, parse_dates=[0],
        index_col=0, date_parser=parser, delimiter='\t',
       decimal=',', thousands='.')
```

```
# Transformar dados para serem estacion rios
raw_values = series.values
diff_values = difference(raw_values, 1)
dataset_diff = diff_values.values

# Transformar dados para serem supervisionados com n mero
   de passos de tempo
dataset = create_dataset(dataset_diff,n_steps)


# Defini  o dos tamanhos de teste e treinamento
train_size = int(dataset.shape[0] * 2/3)
test_size = dataset.shape[0] - train_size

# Dividir o conjunto de dados em duas partes: um
   treinamento e um conjunto de testes.
train, test = dataset[0:train_size], dataset[train_size:]

# Transformar dados para um escala espec fica (
   normaliza  o)
scaler, train_scaled = scale(train)
scaler, test_scaled = scale(test, scaler)

 # Ajuste ao modelo
model = fit_lstm(train_scaled, 1, n_epoch, neurons, n_steps
   , n_features)

# Previs o de todo o conjunto de dados de treinamento para
    o estado de previs o
print('Forecasting Training Data')
predictions_train = []
for i in range(len(train_scaled)):
    # Realiza a previs o
    X, y = train_scaled[i, 0:-1], train_scaled[i, -1]
    yhat = forecast_lstm(model, 1, X, n_steps, n_features)
    # Inverte escala
    yhat = invert_scale(scaler, X, yhat)
    # Inverte diferencia  o
    yhat = inverse_difference(raw_values, yhat, len(
       raw_values)-i)
    # Armazena previs o
    predictions_train.append(yhat)
    expected = raw_values[ i+1 ]
    print('Month=%d, Predicted=%f, Expected=%f' % (i+1,
       yhat, expected))

# Relat rio de performance
```

```
    rmse_train = sqrt(mean_squared_error(raw_values[1:len(
       train_scaled)+1], predictions_train))
    print('Train RMSE: %.3f' % rmse_train)

    # Previs es para gr fico
    predictions = []

    #Valida  o passo a passo nos dados de teste
    print('Forecasting Testing Data')
    train_copy = np.copy(train_scaled)
    predictions_test = list()
    for i in range(len(test_scaled)):
        # update model
        if i > 0:
            update_model(model, train_copy, 1, updates)
        # Realiza previs o
        X, y = test_scaled[i, 0:-1], test_scaled[i, -1]
        yhat = forecast_lstm(model, 1, X, n_steps, n_features)
        # Inverte escala
        yhat = invert_scale(scaler, X, yhat)
        # Inverte diferencia  o
        yhat = inverse_difference(raw_values, yhat, len(
           test_scaled)+1-i)
        # Armazena previs o
        predictions_test.append(yhat)
        # adiciona ao conjunto de treinamento
        train_copy = concatenate((train_copy, test_scaled[i,:].
           reshape(1, -1)))
        expected = raw_values[len(train) + i + 1]
        print('Month=%d, Predicted=%f, Expected=%f' % (i+1,
           yhat, expected))

        # Predictions para o gr fico
        predictions.append(yhat)

# Realizar previ es est ticas (Sem atualiz  o do modelo)
#    print('Forecasting Test Data')
#    # Predictions para gr fico
#    predictions = []
#    # Valida  o passo a passo nos dados de teste
#    predictions_test = []
#    for i in range(len(test_scaled)):
#        # Fazer previs o em uma etapa
#        X, y = test_scaled[i, 0:-1], test_scaled[i, -1]
#        yhat = forecast_lstm(model, 1, X, n_steps, n_features)
#        #yhat = y
#        # Inverte escala
#        yhat = invert_scale(scaler, X, yhat)
```

```
#         # Inverte diferencia  o
#         yhat = inverse_difference(raw_values, yhat, len(
   test_scaled)+1-i)
#
#         # Armazena previs o
#         predictions_test.append(yhat)
#         expected = raw_values[len(train) + i + 1]
#         print('Month=%d, Predicted=%f, Expected=%f' % (i+1,
   yhat, expected))
#
#         # Predictions para o gr fico
#         predictions.append(yhat)

    # # Relat rio de performance
    rmse_test = sqrt(mean_squared_error(raw_values[-len(
       test_scaled):], predictions_test))
    print('Test RMSE: %.3f' % rmse_test)

    # Novas previs es
    print('Forecasting New Data')
    new_predictions = []
    raw_values_p = raw_values.copy()

    for i in range(n_previsao):

        diff_values_p = difference(raw_values_p, 1)

        dataset_diff_p = diff_values_p.values
        dataset_diff_p = dataset_diff_p.reshape(-1, 1)

        dataset_scaled_p = scaler.fit_transform(dataset_diff_p)
        dataset_p = create_dataset(dataset_scaled_p, n_steps)

        if i > 0:
            update_model(model, dataset_p, 1, updates)

        #Busca os ultimos n_steps (look_back)
        X = dataset_scaled_p[-n_steps:]

        yhat = forecast_lstm(model, 1, X, n_steps, n_features)

        yhat = invert_scale(scaler, X, yhat)

        yhat = inverse_difference(raw_values_p, yhat, 1)

        yhat = yhat.flatten()[0]

        raw_values_p = np.append(raw_values_p, yhat)
```

```
    print('Month=%d, Predicted=%f' % (i + 1, yhat))

    new_predictions.append(yhat)

    #Previs es para o gr fico
    predictions.append(yhat)


#Calcula crescimento entre o  ltimo  dado observado e a
    ltima  previs o
last_actual = expected[-1]
print(" ltimo  Valor(Dados):  %.2f" % last_actual)
last_prediction = new_predictions[-1]
print(" ltimo  Valor(Previs o): %.2f" % last_prediction)
pct_pts = ((last_prediction-last_actual)/last_actual)*100
print("Crescimento: %.2f%%" % pct_pts)

# Imprime gr fico
plt.suptitle('CEP ' + cep + ' - ' + nome_modelo, fontsize
   =20)

#plt.figure(figsize=(13,7))
#plt.xticks(np.arange(0, len(train_scaled) + len(
   test_scaled) + n_previsao, step=12.0))

plt.ylabel('Pre os')
plt.xlabel('Per odo')
plt.plot([x for x in raw_values], color='blue', label='
   Pre os')
plt.plot([None for _ in range(len(train_scaled) + 1)] +
         [x for x in predictions], color='orange', label =
             'Previs o de Teste')
plt.plot([None for _ in  range(len(train_scaled) + len(
   test_scaled) + 1)] +
         [x for x in new_predictions], color='red', label
            ='Previs o')
plt.legend(loc='upper center', bbox_to_anchor=(0.5, 1.05),
   ncol=3, fancybox=True, shadow=True)
plt.grid()

ax = plt.gca()
plt.text(0.5, 0.05, 'Crescimento=%.2f%%, RMSE=%.4f' % (
   pct_pts, rmse_test), fontsize=9, ha='center', va='center
   ', transform = ax.transAxes)
plt.savefig(cep + '_' + numero_modelo + '.png')
plt.show()
```

```
    # Busca a  ltima  data e para a partir da  calcular as
      datas de previs es futuras
    ultima_data = datetime.date(series.index[-1])

    rng_datas_previsao = pd.date_range(ultima_data, periods=
      n_previsao + 1, freq='MS', closed='right')
    datas_previsao = pd.DataFrame({'data': rng_datas_previsao})
    datas_previsao.insert(0, 'id', datas_previsao.index)

    dados_previsao = pd.DataFrame(new_predictions,columns=['
      valor'])
    dados_previsao.insert(0, 'id', dados_previsao.index)
    dados_previsao = pd.merge(datas_previsao, dados_previsao,
      on='id')

    saida = pd.DataFrame(dados_previsao, columns=['data','valor
      '])
    saida['data'] = saida['data'].dt.strftime('%d/%m/%Y')
    np.savetxt(cep + 'R_' + numero_modelo + '.csv',saida,
      delimiter =';',fmt='%s;%.2f',header='data;valor')

    return rmse_test

# Fun  o principal para carga de Ceps
def main():

    # Marca in cio do processamento para contagem do tempo
    start_time = time.time()

    # N mero de passos
    n_steps = 1

    # Nome do Modelo
    nome_modelo = 'Modelo um passo a frente'
    numero_modelo = '1' # '1' ou '2'

    # Arquivos
    arquivo_rmse = 'rmse_modelo_1.csv'
    arquivo_resumo = 'resumo_modelo_1.csv'
    arquivo_img_resumo = 'resumo_modelo_1.png'

     # Remove arquivo de rmse
    if os.path.exists(arquivo_rmse):
        os.remove(arquivo_rmse)


    ceps = ['700','702','703','706','707','710',
            '711','712','715','716','717','718',
```

```
        '719','720','721','722','723','724',
        '725','726','727','730','731','732',
        '733','734']

#ceps = ['700']

#----------------MODELO 1
   ------------------------------------------------

error_scores = list()

for cep in ceps:
    rmse = principal(cep, nome_modelo, numero_modelo,
       n_steps)
    error_scores.append(rmse)

# Resumo resultdos
results = pd.DataFrame()
results['rmse'] = error_scores
print(results.describe())
results.describe().to_csv(arquivo_resumo)
results.boxplot()
plt.suptitle(nome_modelo + ' (RMSE)', fontsize=20)
plt.savefig(arquivo_img_resumo)
plt.show()

#----------------MODELO 2
   ------------------------------------------------

# Nome do Modelo
nome_modelo = 'Modelo janela m vel'
numero_modelo = '2'
n_steps = 12

# Arquivos
arquivo_rmse = 'rmse_modelo_2.csv'
arquivo_resumo = 'resumo_modelo_2.csv'
arquivo_img_resumo = 'resumo_modelo_2.png'


error_scores = list()

for cep in ceps:
    rmse = principal(cep, nome_modelo, numero_modelo,
       n_steps)
    error_scores.append(rmse)

# Resumo resultdos
```

```
    results = pd.DataFrame()
    results['rmse'] = error_scores
    print(results.describe())
    results.describe().to_csv(arquivo_resumo)
    results.boxplot()
    plt.suptitle(nome_modelo + ' (RMSE)', fontsize=20)
    plt.savefig(arquivo_img_resumo)
    plt.show()


    #----------------FIM
       ------------------------------------------------

    elapsed_time = time.time() - start_time
    print('Tempo : ' + time.strftime("%H:%M:%S", time.gmtime(
       elapsed_time)))

    # Emite sinal sonoro avisando do t rmino do processamento
    frequency = 1500  # Set Frequency To 2500 Hertz
    duration = 300  # Set Duration To 1000 ms == 1 second
    winsound.Beep(frequency, duration)

if __name__ == '__main__':
    main()
```

7 RESULTADOS GERAIS

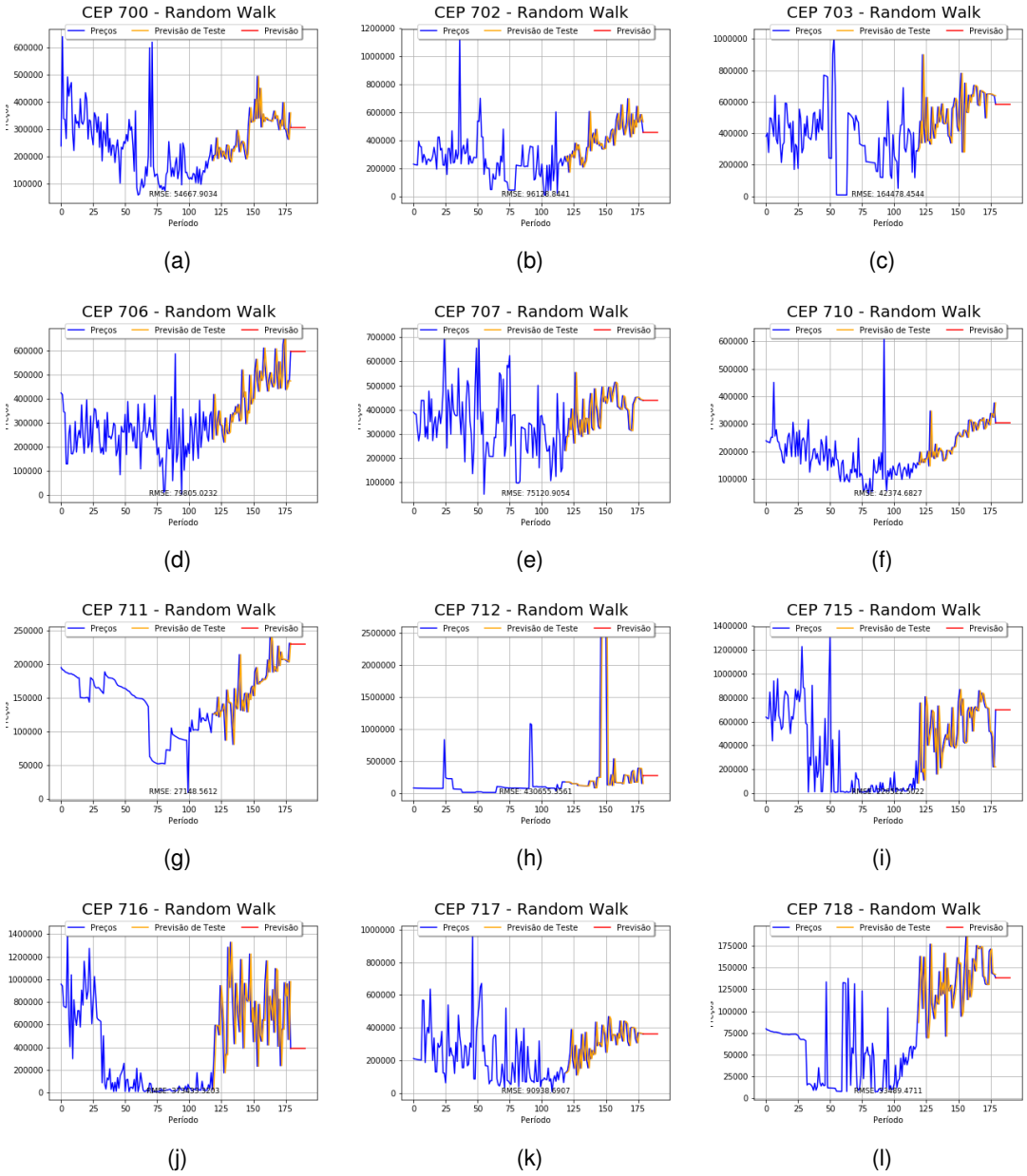


Figura 7.1 – Modelo Random Walk (CEPs 700-718).

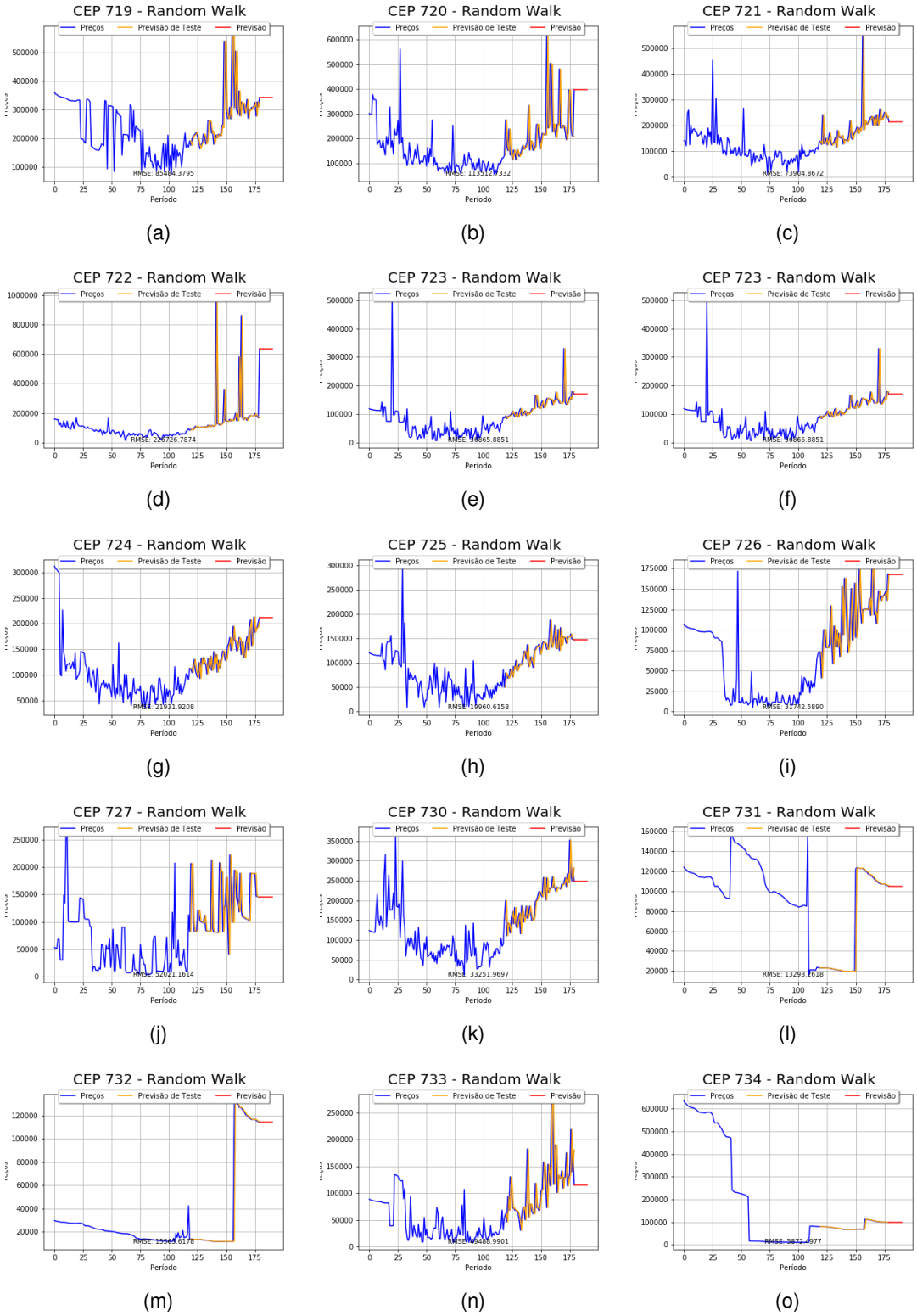


Figura 7.2 – Modelo Random Walk (CEPs 719-734)

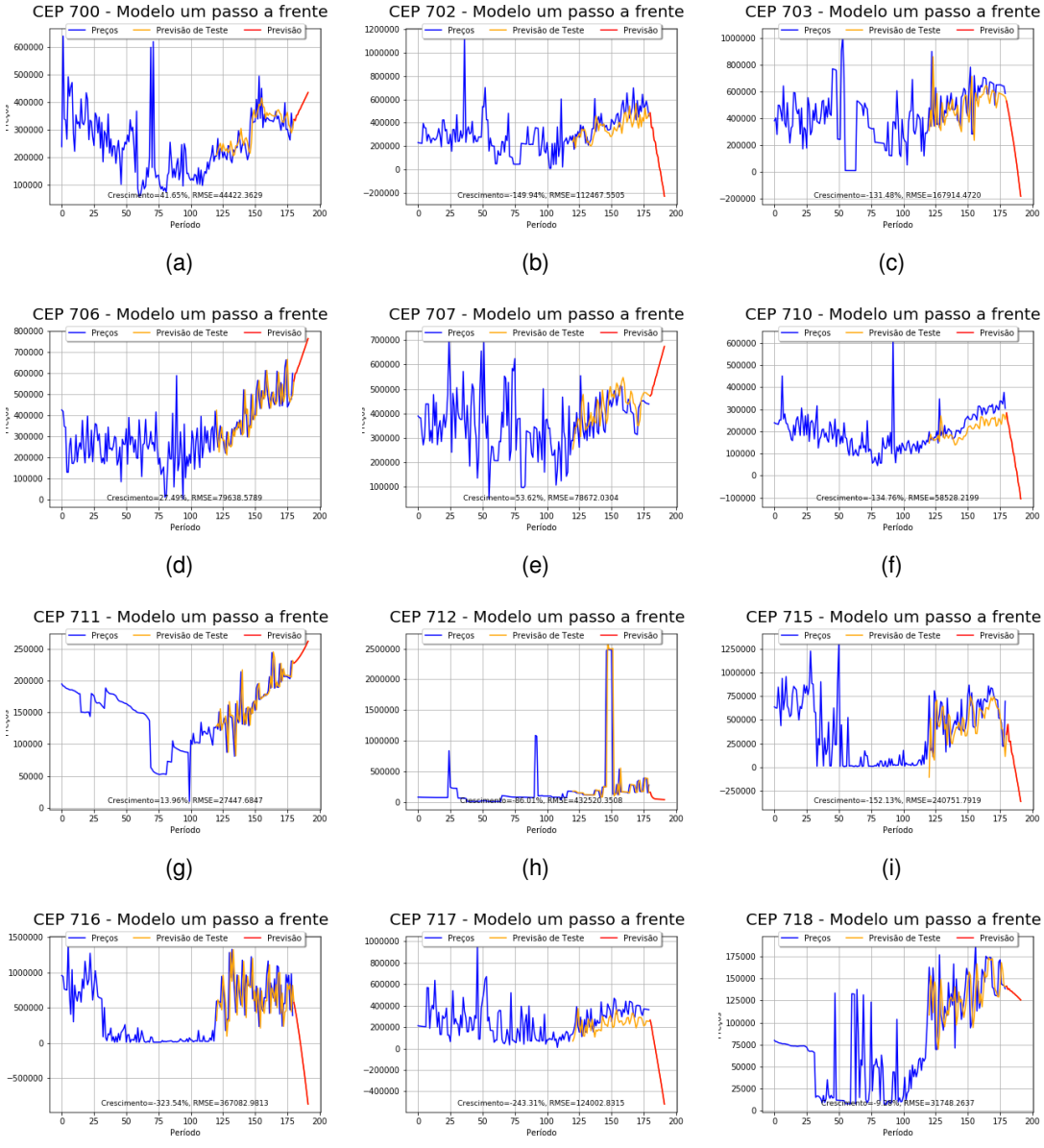


Figura 7.3 – Modelo 1 - Um passo a frente (CEPs 700-718).

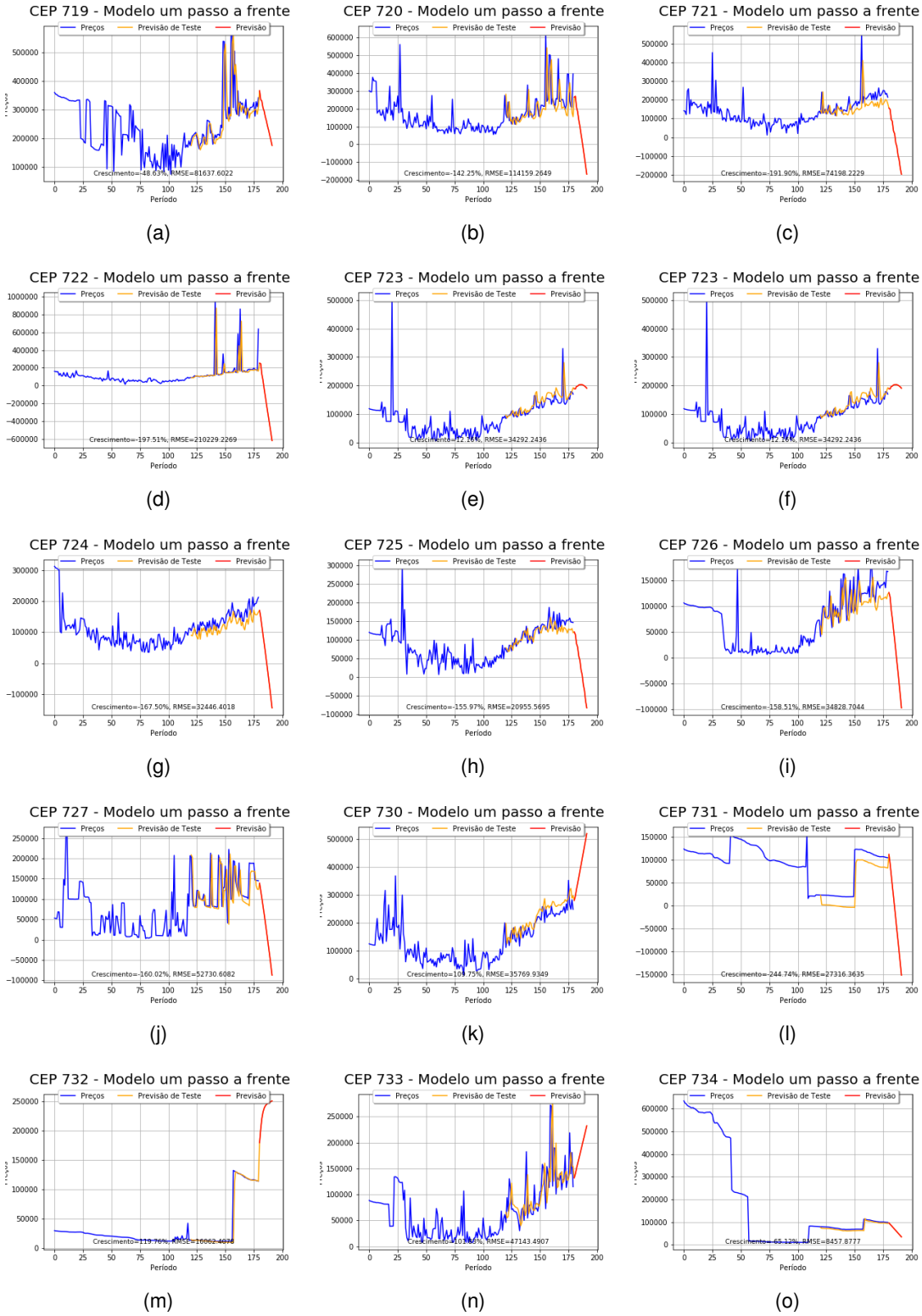


Figura 7.4 – Modelo 1 - Um passo a frente (CEPs 719-734)

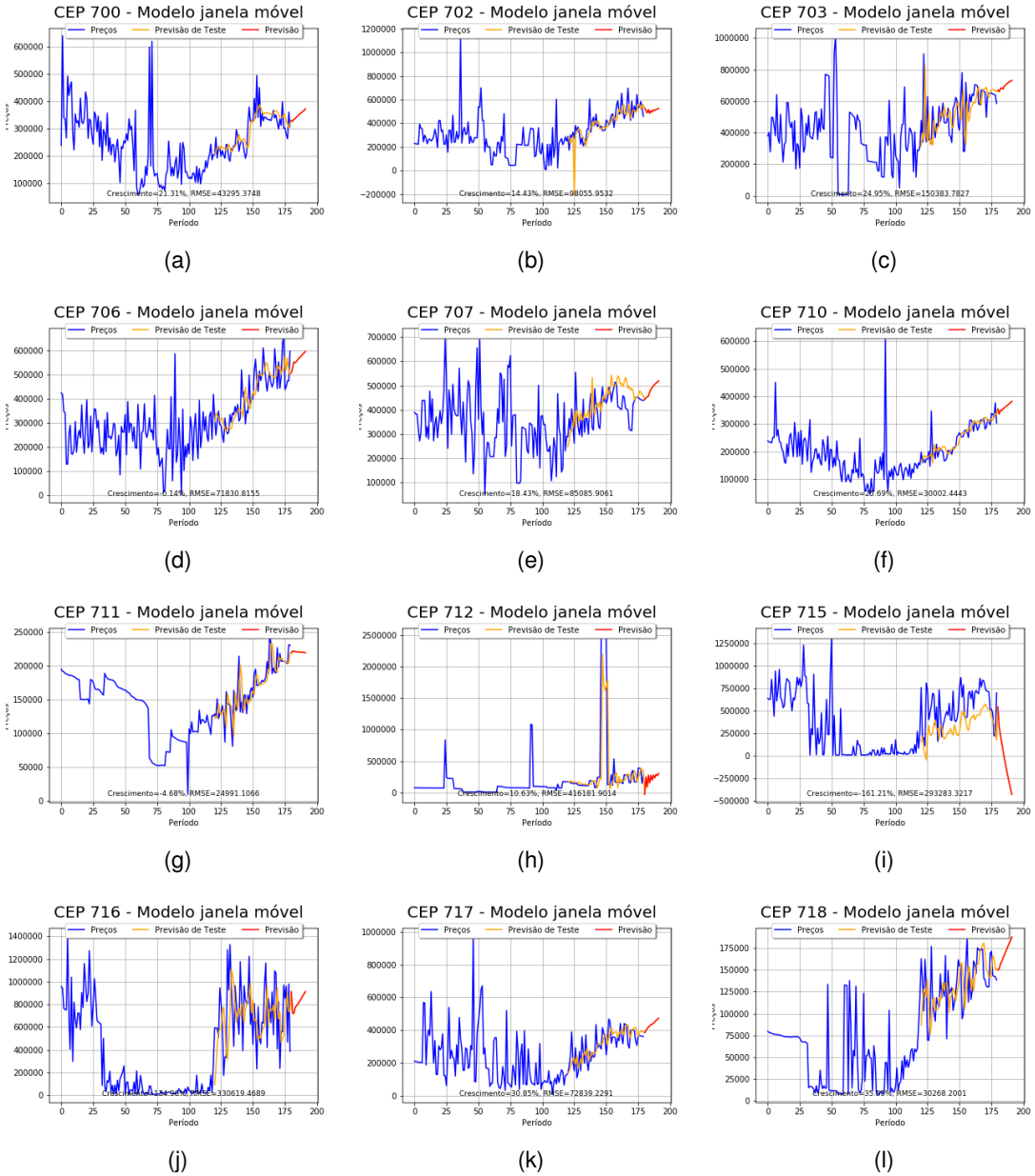


Figura 7.5 – Modelo 2 - Janela Móvel (CEPs 700-718).

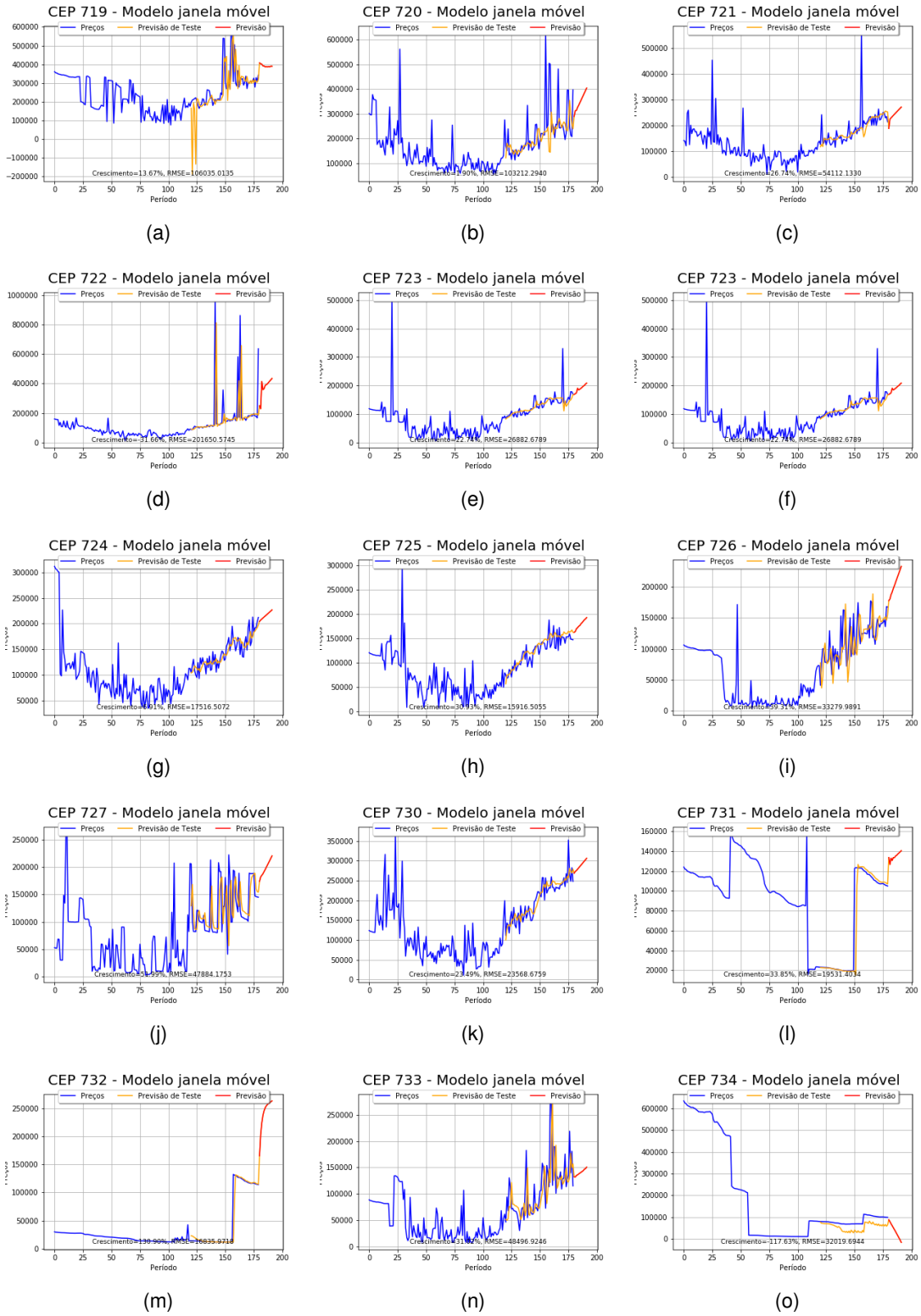


Figura 7.6 – Modelo 2 - Janela Móvel (CEPs 719-734).